Geometric Methods in Learning and Memory

Dimitri Novytskyi

No D'ORDRE:

THESE

présentée devant

L'UNIVERSITE PAUL SABATIER de TOULOUSE

en vue de l'obtention

du Doctorat de L'UNIVERSITE

Spécialité: *Mathématiques Appliquées*

par

Dmytro NOVYTSKYY

Sujet :

Méthodes géométriques pour la mémoire et l'apprentissage

Date de soutenance : 13 Juillet 2007

Membres de Jury :

Tatiana. AKSENOVA	Rapporteur	Directeur de Recherche, IASA, Kiev
Jean-Pierre DEDIEU	Co-Directeur de Thèse	Professeur à l'Université Paul Sabatier, MIP, Toulouse
Mohammed MASMOUDI		Professeur à l'Université Paul Sabatier, MIP, Toulouse
Jean-Marie MORVAN	Rapporteur	Professeur de Mathématiques Université Claude Bernard Lyon 1
Michael KUSSUL		Directeur de Recherche, IMMS, Kiev
Jean-Claude YAKOUBSOHN	Président du jury	Professeur à l'Université Paul Sabatier ,.Directeur adjoint, MIP, Toulouse

Présentation.

La thèse présentée ici est une thèse en co-tutelle dirigée par Naum Shor (Institute of Cybernetics, Kiev) puis Alexander Reznik (Institute of Mathematical Machines and Systems, Kiev) et par Jean-Pierre Dedieu (Institut des Mathematiques de Toulouse).

Ce travail a bénéficié d'une bourse du gouvernement français attribuée par l'Ambassade de France à Kiev.

Naum Shor est décède prématurément le 26 février 2006. Nous lui dédions ce travail, in memoriam.

Geometric methods of learning

Introduction

In many problems of (supervised and unsupervised) learning, pattern recognition, and clustering there is a need to take in account the internal (intrinsic) structure of the underlying space, which is not necessary Euclidean. Such tasks include associative memories [1], independent component analysis [2], signal processing [3], etc.

Spaces emerging in these problems can be Riemannian spaces, in particular Lie groups and homogeneous spaces, or metric spaces without any Riemannian structure etc.

In recent publications [7, 8 etc.] we find several types of such problems posed for neural networks, Kalman–like filters, and blind signal separation… Amari (1998) [5] showed that gradient methods on manifolds are suitable for feedforward-like neural networks. Fiori [6] proposed several types of learning algorithms on homogeneous Riemannian manifolds (like orthogonal groups, Stiefel and Grassmann manifolds) for independent component analysis (ICA), and blind signal separation. Celledoni and Fiori [9] use rigid-body dynamical model for learning. In [4] Lie groups are used for recognition and model identification.
In the present thesis we present several methods on Riemannian spaces (calculation of geodesics, Newton methods, conjugate-gradient methods), then we develop some algorithms for learning on Riemannian and metric spaces, and provide experimental evidence on academic tests and real-life problems.

Chapter 1 is devoted to the Newton method and geodesic calculations in Riemannian manifolds. Geodesic computation is itself a nontrivial problem. The traditional way of geodesic computation is based on local coordinates and Christoffel symbols. Unfortunately, this way is difficult to implement because there is no universal procedure of chart changing.
We propose here a technique based on global coordinates in Euclidean spaces (R^n), where the manifold is embedded. In this space we write the Hamilton equations describing the trajectory of a free particle attached to the manifold. Classical mechanic shows that such trajectories are precisely geodesics on this manifold. Then we compute these trajectories using symplectic Runge-Kutta methods. The corresponding paper is published in *Journal of Complexity* vol 21. (2005) pp. 487-501.

In **Chapter 2** we study generalized averaging on Grassmann manifolds. Then we show that the space of pseudo-inverse associative memories with fixed dimension is isomorphic to a Grassmann manifold, and develop an algorithm of unsupervised learning and clustering for Hopfield-like Neural Associative memory. This procedure enables us to endow the associative memory with ability of data generalization. After the synthesis of associative memory containing generalized data, cluster centers are retrieved using procedure of associative recall with random starts. The algorithm is tested for artificial data problem of unsupervised image recognition and for a set of real-life small images (MNIST database). The corresponding papers are submitted to *Neural Networks* (2007) and published in *Proc. ESANN 2005,* Bruges, Belgium, April, 27-29.

In **Chapter 3** kernel methods of neural associative memory are presented. In combination of the algorithm from Chapter 3 they provide an unsupervised learning for wider class of tasks. Kernel methods, including support vector machines (SVM), least square SVM, and some more techniques, are based on implicit extension of a feature space, than only scalar product in the new (high or even infinite-dimensional) space is used. Currently, kernel machines are mostly applied to the tasks of pattern recognition and classification. Here we expand their use to Hopfield-like neural associative memories and use them for recognition and associative recall. The corresponding paper is published in *Proc. IJCNN'04*, Budapest, Hungary, July 25-28.

In **Chapter 4** there is a method of generalized averaging in the space of signal trajectories in the phase space. Such a space can also be treated as an orbit space with respect to infinite Lie group of time changes:

$$t \to \theta(t)$$

where θ is a monotonic twice differentiable function. This approach is used for constructing a nonlinear algorithm for suppression of artifacts of deep brain stimulation from records of neural activity. The corresponding paper is submitted to *Neural Computation* (2007).

In **Chapter 5** some of described approaches are used for a real-life problem: a system of odor recognition (electronic nose). We provide experimental evidence that our proposed methods are competitive in comparison with classical techniques and often outperform them. Corresponding

papers are published in *Sensors and Actuators B,* vol. 106 (2005), pp. 158-163 and *Proc. of Int. Conf. on Neural Information Processing,* Singapore 2002.

To conclude this brief description of the manuscript I would say that our point of view was to treat all the aspects of the considered problems: from real life to the mathematical model then, via a good algorithmic, to the various aspects of the implementation and then back to real life. Our objective was to design algorithms which respect the geometric structure of the problem.

This thesis has been elaborated both in Kiev (Institute of Cybernetics of NASU) under the direction of Naum Shor then Alexander Reznik and in Toulouse (Institut de Mathématiques) under the direction of Jean-Pierre Dedieu. It was supported by a grant from the French government.

I would like here to thank my advisors for directing this multidisciplinary work, Luca Amodei for his suggestions on symplectic methods, Tamara Bardadym and Petro Stetisiuk for their advices on nonsmooth optimization, Jean-Claude Yakoubson and Mohammed Masmoudi for collaboration in Neural Networks, optimization, and finding zeros on manifolds, Jean-Marie Morvan and Tatiana Aksenova for reviewing this thesis, as well all my collegues from Dept. of Neural Technologies and Institute of Cybernetics, Kiev and the Laboratory MIP, Toulouse for many helpful discussions, the pleasant and supportive atmosphere, and hospitality.

Refereces

1. Kohonen, Teuvo *Self-organization and associative memory*. Third edition. Springer Series in Information Sciences, 8. Springer-Verlag, Berlin, 1989. xvi+312 pp.
2. S.-I. Amari and S. Fiori. Editorial: Special issue on “Geometrical Methods in Neural Networks and Learning" Neural Networks, *Neural Networks, Vol. 18 (2005),*
3. S.Fiori. Blind signal processing by the adaptive activation function neurons *Neural Networks, Volume 13, Issue 6, July 2000*
4. G **Arnold**, K Sturtz, V Velten. Lie group analysis in object recognition**.** *Proc. DARPA Image Understanding Workshop*, 1997

5. S Amari. Natural Gradient Works Efficiently in Learning - *Neural Computation*, 1998 - MIT Press
6. S. Fiori. Quasi-Geodesic Neural Learning Algorithms over the Orthogonal Group: A Tutorial. *Neural Networks, Vol. 18 (2005),*
7. S. Fiori. Formulation and Integration of Learning Differential Equations on the Stiefel Manifold. *IEEE Trans. Neural Networks* **Vol. 16** No 5 (2005)
8. Wang Shoujue, Lai Jiangliang Geometrical learning, descriptive geometry, biomimetic pattern recognition. *Neurocomputig* (2005)
9. E Celledoni, S Fiori. Neural learning by geometric integration of reduced 'rigid-body' equations.- *Journal of Computational and Applied Mathematics*, 172 (2004) 247–269

Chapter 1

Symplectic Methods for the Approximation of the Exponential Map and the Newton Iteration on Riemannian Submanifolds.

Jean-Pierre Dedieu* Dmitry Nowicki†

June 15, 2004

1 Introduction.

Let V be a $p-$dimensional Riemannian real complete manifold. In this paper we study computational aspects of the Newton method for finding zeros of smooth mappings $f : V \to \mathbb{R}^p$. The Newton operator is defined by

$$N_f(x) = \exp_x(-Df(x)^{-1}f(x)) \tag{1}$$

Here $\exp_x : T_xV \to V$ is the exponential map, which "projects" the tangent space at x on the manifold. The Newton method has two important properties: fixed points for N_f correspond to zeros for f and the convergence of the Newton sequences ($x_0 = x$ and $x_{k+1} = N_f(x_k)$) is quadratic for any starting point x in a neighborhood of a nonsingular zero.

When $V = \mathbb{R}^n$, the exponential map is just a translation: $\exp_x(u) = x + u$ and the Newton operator has the usual form:

$$N_f(x) = x - Df(x)^{-1}f(x)$$

but for a general manifold this is no more true. Except for some cases the exponential map has no analytic expression and we have to compute it numerically: this is the main subject of this paper.

Newton method for maps or vector fields defined on manifolds has already been considered by many authors: Shub 1986 [29] defines Newton's method for

*MIP. Département de Mathématique, Université Paul Sabatier, 31062 Toulouse cedex 04, France (dedieu@mip.ups-tlse.fr).

†MIP. Département de Mathématique, Université Paul Sabatier, 31062 Toulouse cedex 04, France (nowicki@mip.ups-tlse.fr).

the problem of finding the zeros of a vector field on a manifold and uses retractions to send a neighborhood of the origin in the tangent space onto the manifold itself. Udriste 1994 [35] studies Newton's method to find the zeros of a gradient vector field defined on a Riemannian manifold; Owren and Welfert 1996 [27] define Newton iteration for solving the equation $f(x) = 0$ where f is a map from a Lie group to its corresponding Lie algebra; Smith 1994 [34] and Edelman-Arias-Smith 1998 [10] develop Newton and conjugate gradient algorithms on the Grassmann and Stiefel manifolds. Shub 1993 [30], Shub and Smale 1993-1996 [31], [32], [33], see also, Blum-Cucker-Shub-Smale 1998 [4], Malajovich 1994 [22], Dedieu and Shub 2000 [7] introduce and study the Newton method on projective spaces and their products. Another paper on this subject is Adler-Dedieu-Margulies-Martens-Shub 2001 [3] where qualitative aspects of Newton method on Riemannian manifolds are investigated for both mappings and vector fields. This paper contains an application to a geometric model for the human spine represented as a $18-$tuple of 3×3 orthogonal matrices. Recently Ferreira-Svaiter [11] give a Kantorovich like theorem for Newton method for vector fields defined on Riemannian manifolds and Dedieu-Malajovich-Priouret [6] study alpha-theory for both mappings and vector fields.

The computation of the exponential map depends mainly on the considered data structure. In some cases the exponential is given explicitely (Euclidean or projective spaces, spheres ...) or may be computed via linear algebra packages (the orthogonal group, Stiefel or Grassmann manifolds [10], [34]). The classical description uses local coordinates and the second order system which gives the geodesic curve $x(t)$ with initial conditions $x(0) = x$, and $\dot{x}(0) = u$:

$$\ddot{x}_i(t) + \sum_{j,k} \Gamma^i_{jk} \dot{x}_j(t) \dot{x}_k(t) = 0, \quad 1 \leq i \leq n,$$

$$x(0) = x, \quad \dot{x}(0) = u.$$

In these equations Γ^i_{jk} are the Christoffel symbols and the exponential is equal to $\exp_x(u) = x(1)$, see Do Carmo [9] or others textbooks on this subject: Dieudonné [8], Gallot-Hulin-Lafontaine [13], Helgason [17]. Such an approach is used by Noakes [25] who considers the problem of finding geodesics joining two given points. We notice that the computation of local coordinates and of the Christoffel symbols may be itself a very serious problem and depends again on the data structure giving the manifold V.

In [5] Celledoni and Iserles consider the approximation of the exponential for finite dimentional Lie groups contained in the general linear group using splitting techniques. Munthe-Kaas-Zanna [36] approximate the matrix exponential by the use of a generalized polar decomposition. See also Munthe-Kaas-Quispel-Zanna [36] for the generalized polar decomposition on Lie groups, Krogstad-Munthe-Kaas-Zanna [21] and Iserles-Munthe-Kaas-Nørset-Zanna [19].

In this paper we concentrate our efforts on submanifolds. Let $F : U \to \mathbb{R}^m$ be a C^2 map where $U \subset \mathbb{R}^n$ is open. Let V denote its zero set: $V = F^{-1}(0) \subset U \subset \mathbb{R}^n$. We suppose that $DF(x) : \mathbb{R}^n \to \mathbb{R}^m$ is onto for each $x \in U$. In that case, V is a C^2 submanifold contained in $\mathbb{R}^n$ and its dimension is equal to $p = n-m$. V is equipped with the Riemannian structure inherited from $\mathbb{R}^n$: the scalar product on T_xV is the restriction of the usual scalar product in $\mathbb{R}^n$. This case is particularly important in optimization theory when V, the set of feasible points, is defined by equality constraints. In this framework, to compute the geodesic curves with initial value conditions, we take a mechanical approach: a geodesic is the trajectory of a free particle attached to the submanifold V, see Abraham-Marsden [1] or Marsden-Ratiu [23]. We give a first description of this trajectory in terms of Lagrangian equations and then, via an optimal control approach and Pontryagin's maximum principle, in terms of Hamiltonian equations. Our numerical methods are based on this last system: we use symplectic methods to solve it (second, fourth or sixth order Gauss method).

We are now able to compute the Newton operator attached to a system of equations defined on V, say $f : V \to \mathbb{R}^{n-m}$. The last section is devoted to numerical examples. We compare this Riemannian Newton method (called here GNI for "Geometric Newton Iteration") with the usual Euclidean Newton method (called CNI for "Classical Newton Iteration") which solves the extended system $f(x) = 0$, and $F(x) = 0$ with $x \in \mathbb{R}^n$. Both methods, for these examples, give comparable results with a smaller number of iterates for the GNI and a slightly better accuracy for the CNI.

Other numerical methods for problems posed on Riemannian manifolds requiere the computation of the exponential map. This will be the purpose of a second paper. We thanks here Luca Amodei for valuable discussions about this symplectic approach.

2 The equations defining the geodesics.

The exponential map $\exp_x : T_xV \to V$ is defined in the following way: for $x \in V$ and $u \in T_xV$ let $x(t)$, $t \in \mathbb{R}$, be the geodesic curve such that $x(0) = x$ and $\dot{x}(0) = \frac{dx(t)}{dt}|_{t=0} = u$. Then $\exp_x(u) = x(1)$. Let us denote by N_xV the normal space at x. We have

$$T_xV = \text{Ker } DF(x) \text{ and } N_xV = (T_xV)^\perp = \text{Im } DF(x)^*.$$

This geodesic is characterized by the following system:

$$\begin{array}{l} x(t) \in V, \\ \ddot{x}(t) \in N_{x(t)}V, \\ x(0) = x,\ \dot{x}(0) = u. \end{array} \tag{2}$$

We introduce a Lagrange multiplier $\lambda(t) \in \mathbb{R}^m$ so that the system 2 becomes

$$\begin{array}{l} F(x(t)) = 0, \\ \ddot{x}(t) = -DF(x(t))^* \lambda(t), \\ x(0) = x,\ \dot{x}(0) = u. \end{array} \tag{3}$$

This geodesic curve may be interpreted as the trajectory of a free particle attached to V. Using the formalism of Lagrangian mechanics, see Marsden-Ratiu [23] section 8.3, we notice that this system is given by the Euler-Lagrange equation associated with the following Lagrangian:

$$\mathcal{L}(x, \dot{x}, \lambda) = \frac{1}{2}\|\dot{x}\|^2 - \sum_{i=1}^{m} \lambda_i F_i(x) \tag{4}$$

that is

$$\begin{array}{l} F(x(t)) = 0, \\ \dfrac{d}{dt}\dfrac{\partial \mathcal{L}}{\partial \dot{x}} = \dfrac{\partial \mathcal{L}}{\partial x}, \\ x(0) = x,\ \dot{x}(0) = u. \end{array} \tag{5}$$

Definition 2.1 *For a linear operator $A : E \to F$ between two Euclidean spaces, we denote by $A^\dagger$ its generalized inverse. It is the composition of three maps, $A^\dagger = i \circ B^{-1} \circ \Pi_{Im\ A}$ with $\Pi_{Im\ A}$ the orthogonal projection from F onto Im A, $B : (Ker\ A)^\perp \to Im\ A$ the restriction of A, $i : (Ker\ A)^\perp \to E$ the canonical injection.*

The operator $AA^\dagger$ is equal to the orthogonal projection $F \to \text{Im } A$ and $A^\dagger A$ is the orthogonal projection $E \to (\text{Ker } A)^\perp$. When A is onto one has $A^\dagger = A^*(AA^*)^{-1}$ and $AA^\dagger = \text{id}_F$ while, when A is injective, $A^\dagger = (A^*A)^{-1}A^*$ and $A^\dagger A = \text{id}_E$.

Proposition 2.1 *For any $x \in V$ and $u \in T_xV$ the system 3 is equivalent to:*

$$\begin{array}{l} \ddot{x}(t) = -DF(x(t))^\dagger D^2F(x(t))(\dot{x}(t), \dot{x}(t)), \\ \lambda(t) = \left(DF(x(t))DF(x(t))^*\right)^{-1} D^2F(x(t))(\dot{x}(t), \dot{x}(t)), \\ x(0) = x,\ \dot{x}(0) = u. \end{array} \tag{6}$$

Proof. To obtain 6 from 3 we differentiate two times $F(x(t)) = 0$ so that

$$D^2F(x(t))(\dot{x}(t), \dot{x}(t)) + DF(x(t))\ddot{x}(t) = 0.$$

By 3 we get

$$D^2F(x(t))(\dot{x}(t), \dot{x}(t)) - DF(x(t))DF(x(t))^*\lambda(t) = 0.$$

Since $DF(x(t))$ is onto, $DF(x(t))DF(x(t))^*$ is nonsingular and this gives $\lambda(t)$ and $\ddot{x}(t)$. Conversely, 6 gives

$$\ddot{x}(t) = -DF(x(t))^\dagger D^2F(x(t))(\dot{x}(t), \dot{x}(t)) = -DF(x(t))^*\lambda(t).$$

Moreover

$$DF(x(t))\ddot{x}(t) = -D^2F(x(t))(\dot{x}(t), \dot{x}(t))$$

that is

$$\frac{d^2}{dt^2}F(x(t)) = 0.$$

This gives

$$F(x(t)) = F(x(0)) + DF(x(0))\dot{x}(0) + \frac{1}{2}\int_0^t \frac{d^2}{dt^2}F(x(s))ds = F(x) + DF(x)u + 0 = 0.$$

■

Let us now introduce the Hamilton equations. To obtain them we consider the problem of finding a minimizing geodesic with two given endpoints as the following optimal control problem (see Udriste [35]):

$$\min \int_0^T \|u(t)\|^2 dt$$

subject to the constraints $\dot{x} = u$, $x(0) = x_0$, $x(1) = x_1$, $F(x(t)) = 0$ for every $t \in [0, T]$, where x_0 and x_1 are given points in V. According to Pontryagin's maximum principle, the Hamiltonian for problems like

$$\min \int_0^T f_0(x, u, t)dt$$

subject to the constraints $\dot{x} = f(x, u, t)$, $x(0) = x_0$, $x(1) = x_1$, $F(x(t)) = 0$ for every $t \in [0, T]$, can be written as

$$\mathcal{H}(p, x, \mu) = -f_0 + \langle p, f\rangle + \sum_{i=1}^m \mu_i DF_i(x)\dot{x}.$$

In our case we obtain

$$\mathcal{H}(x, p, \mu) = \langle p, \dot{x}\rangle - \frac{1}{2}\|\dot{x}\|^2 + \sum_{i=1}^m \mu_i DF_i(x)\dot{x}$$

with $p \in \mathbb{R}^n$, $\mu \in \mathbb{R}^m$. The Hamilton equations are

$$\begin{aligned} &\dot{p}(t) = -\frac{\partial \mathcal{H}}{\partial x}(x(t), p(t), \mu(t)), \\ &p(t) = \dot{x}(t) - DF(x(t))^*\mu(t), \\ &\dot{\mu}(t) = -\lambda(t), \quad \mu(0) = 0. \end{aligned} \tag{7}$$

Proposition 2.2 *Let $x \in V$ and $u \in T_xV$ be given. The system 7 is equivalent to*

$$\begin{aligned} &\dot{p}(t) = -\sum_{i=1}^m \mu_i D^2 F_i(x(t))\dot{x}(t), \\ &\dot{x}(t) = \Pi_{T_{x(t)}V} p(t) = \left(id - DF(x(t))^\dagger DF(x(t))\right) p(t), \\ &\mu(t) = -DF(x(t))^{*\dagger} p(t), \\ &x(0) = x, \quad p(0) = u \end{aligned} \tag{8}$$

which is also equivalent to the system 3.

Proof. To obtain 8.1 from 7 we differentiate $\mathcal{H}$ with respect to x to obtain

$$\frac{\partial \mathcal{H}}{\partial x} = \left\langle p, \frac{\partial \dot{x}}{\partial x} \right\rangle - \left\langle \dot{x}, \frac{\partial \dot{x}}{\partial x} \right\rangle + \sum_{i=1}^m \mu_i DF_i(x) \frac{\partial \dot{x}}{\partial x} + \sum_{i=1}^m \mu_i D^2 F_i(x)\dot{x} = \sum_{i=1}^m \mu_i D^2 F_i(x)\dot{x}$$

by 7.2. The two other equations in 8 are obtained from 7.2 by projecting $p(t)$ on Ker $DF(x(t))$ and Ker $DF(x(t))^\perp = \text{Im } DF(x(t))^*$ so that

$$\dot{x}(t) = \Pi_{T_{x(t)V}} p(t)$$

and

$$-DF(x(t))^* \mu(t) = \Pi_{\text{Im } DF(x(t))^*} p(t).$$

Since $DF(x(t))$ is injective we get

$$\mu(t) = DF(x(t))^{*\dagger} DF(x(t))^* \mu(t) =$$

$$-DF(x(t))^{*\dagger} \Pi_{\text{Im } DF(x(t))^*} p(t) = -DF(x(t))^{*\dagger} p(t).$$

To obtain 8.1 from 3 and 6 we differentiate 7 to obtain

$$\dot{p}(t) = \ddot{x}(t) - DF(x(t))^* \dot{\mu}(t) - \sum_{i=1}^m \mu_i D^2 F_i(x(t))\dot{x}(t) =$$

$$\ddot{x}(t) + DF(x(t))^* \lambda(t) - \sum_{i=1}^m \mu_i D^2 F_i(x(t))\dot{x}(t) = -\sum_{i=1}^m \mu_i D^2 F_i(x(t))\dot{x}(t).$$

The initial condition 8.4 is given by

$$p(0) = \dot{x}(0) - DF(x(0))^* \mu(0) = u.$$

To obtain 3 from 8 we differentiate $\dot{x}(t) = p(t) + DF(x(t))^* \mu(t)$ to get

$$\ddot{x}(t) = \dot{p}(t) + \sum_{i=1}^m \mu_i D^2 F_i(x(t))\dot{x}(t) + DF(x(t))^* \dot{\mu}(t) = DF(x(t))^* \lambda(t).$$

By the same equation we get

$$\dot{x}(0) = p(0) + DF(x(0))^* \mu(0) = u.$$

Moreover,

$$DF(x(t))\dot{x}(t) = DF(x(t))\Pi_{\text{Ker } DF(x(t))}p(t) = 0,$$

by integrating we get

$$F(x(t)) = F(x(0)) + \int_0^t DF(x(s))\dot{x}(s)ds = 0$$

and we are done. ■

3 Numerical integration of Hamilton Equations.

3.1 Symplectic Runge-Kutta methods.

To integrate the Hamiltonian system (8) we use symplectic Runge-Kutta methods. We do not use partitioned Runge-Kutta methods like Stormer-Verlet because our Hamiltonnian is not separable. Let us consider an autonomous system:

$$\dot{y} = G(y) : U \to \mathbb{R}^p \tag{9}$$

defined over an open set $U \subset \mathbb{R}^p$. In the case considered here $y = (x, p) \in \mathbb{R}^n \times \mathbb{R}^n$ and $G(x, p)$ is given by (8).

Let us denote by $\Phi_t(y)$ the associated integral flow : $y(t) = \Phi_t(y)$ is the solution of (9) with the initial condition $y(0) = y$. The implicit Runge-Kutta method we have implemented is given by

$$\begin{aligned} y_0 &= y(0) \\ y_{k+1} &= y_k + \tau \sum_{i=1}^{s} b_i G(Y_i) \\ Y_i &= y_k + \tau \sum_{j=1}^{s} a_{ij} G(Y_j),\ 1 \le i \le s. \end{aligned} \tag{10}$$

Here $\tau > 0$ is the given step size, s is a given integer, $Y_1, \ldots, Y_s$ are auxiliary variables , $(a_{ij})_{1 \le i,j \le s}$, $(b_i)_{1 \le i \le s}$ are the coefficients defining the considered method. For our experiments we use Gauss methods of order 2, 4 and 6. The corresponding coefficients are

- Order 2: $s = 1$, $a_{11} = \frac{1}{2}$ and $b_1 = 1$.
- Order 4: $s = 2$,

a_{ij}		
	$\frac{1}{4}$	$\frac{1}{4} - \frac{\sqrt{3}}{6}$
	$\frac{1}{4} + \frac{\sqrt{3}}{6}$	$\frac{1}{4}$
b_i	$\frac{1}{2}$	$\frac{1}{2}$

- Order 6: $s = 3$,

a_{ij}			
	$\frac{5}{36}$	$\frac{2}{9} - \frac{\sqrt{15}}{15}$	$\frac{5}{36} - \frac{\sqrt{15}}{30}$
	$\frac{5}{36} + \frac{\sqrt{15}}{24}$	$\frac{2}{9}$	$\frac{5}{36} - \frac{\sqrt{15}}{24}$
	$\frac{5}{36} + \frac{\sqrt{15}}{30}$	$\frac{2}{9} + \frac{\sqrt{15}}{15}$	$\frac{5}{36}$
b_i	$\frac{5}{18}$	$\frac{4}{9}$	$\frac{5}{18}$

see Hairer-Norsett-Wanner [16] or Sanz-Serna-Calvo [28] about these methods.

Let us denote $\psi_t : \mathbb{R}^p \to \mathbb{R}^p$ which outputs y_{k+1} in terms of y_k. Let us consider again the case of (8) with $y = (x, p)$. The properties of these methods are the following

- They are symplectic i. e.

  $$\omega^2(\psi_t(y)) = \omega^2(y)$$

  for any Hamiltonian system and for any $\tau > 0$ where ω^2 is the differential 2-form

  $$\omega^2 = \sum_{i=1}^{n} dx_i \wedge dp_i. \tag{11}$$

To solve equations 10 we have chosen a successive approximation scheme. These iterations are convergent when the following inequality

$$\|DG\|\|A\|\tau < 1$$

is satisfied. The norm of DG could be estimated by a direct derivation of the right-hand side of system 7. This lead to the following inequality

$$\|DG\| \leq \|D^2F\|^2\|(DFDF^*)^{-1}\| + \|D^2F\|^2\|(DFDF^*)^{-1}\|^2\|DF\|^2 +$$

$$\|DF\|\|D^3F\|\|(DFDF^*)^{-1}\|.$$

Let ν denotes the index of internal iteration inside the k-th step. For a given tolerance tol our termination criterion is

$$\|Y^{\nu+1} - Y^{\nu}\| \leq \tau.tol.$$

In our experiments we have chosen $tol \approx 10^{-8}$. Further decreasing of the tolerance did not lead to better accuracy of the geometric Newton method.

3.2 Backward error analysis.

We apply the backward error analysis techniques from Hairer-Lubich [15] and Hairer [14] to the case of the system (8) integrated by symplectic methods. We show that the computed points x_k ($y_k = (x_k, p_k)$ are arbitrarily close to the geodesic corresponding to a nearby Riemannian structure and the same initial conditions as in the exact problem. We also estimate the distance between these two Riemannian distances. More precisely

Theorem 3.1 *Let $V \subset \mathbb{R}^n$ be a Riemannian submanifold defined by the equation $F(x) = 0$, where $F : U \subset \mathbb{R}^n \to \mathbb{R}^m$ is analytic on a certain neighbourhood of D in $\mathbb{C}^n$. Let I be a symplectic numerical integrator of order r with a sufficiently small step size $\tau > 0$. Let us denote by $g(x)$ the $n \times n$ positive definite matrix defining the Riemannian metric at $x \in V$. Then V can be endowed with a new Riemannian metric $\tilde{g}(x, \tau)$ such that*

1. $$\|g(x) - \tilde{g}(x, \tau)\| = O(\tau^{r+1}), \tag{12}$$

2. *There exists $\tau^* > 0$ such that for any initial condition*

$$x(0) = x_0 \in V, \ \dot{x}(0) = u_0 \in T_{x_0}V \tag{13}$$

we have

$$\|x_k - \tilde{x}(k\tau)\| = O(\exp(-\frac{\tau^*}{2\tau})) \tag{14}$$

for any k such that $k\tau \leq T = \tau exp\,(\tau^*/2\tau)$, where x_k is the numerical solution provided by the integrator and $\tilde{x}(t)$ is the exact geodesic associated with the metric $\tilde{g}(x, \tau)$ and the same initial conditions (13).

Proof. We apply the Corollary 6 from Hairer-Lubich 1997 [15] to our system. See also Hairer [14] where constrained Hamiltonian systems are considered. From this corollary, we get a Hamiltonian

$$\tilde{H}(x, p, \tau) = H(x, p) + O(\tau^{r+1}) \tag{15}$$

such that its trajectories satisfy the estimate (14). The Riemannian metric $\tilde{g}$ is built from the kinetic energy of this new Hamiltonian (see Arnold [2], chapter 9). This metric satisfies the inequality (12). The trajectories of the Hamiltonian 15 are the geodesics of this metric. □

4 The Newton operator

How do we compute a Newton step ? Let us first recall the geometric context.

Let $F : U \to \mathbb{R}^m$ be a C^2 map where $U \subset \mathbb{R}^n$ is open. Let V denote its zero set: $V = F^{-1}(0) \subset U \subset \mathbb{R}^n$ and let $f : U \to \mathbb{R}^{n-m}$ be given, as smooth as necessary. We also denote by $\hat{f}$ its restriction to V.

To compute the Newton operator

$$N_{\hat{f}}(x) = \exp_x(-D\hat{f}(x)^{-1}\hat{f}(x))$$

we need the derivative $D\hat{f}(x) : T_xV \to \mathbb{R}^{n-m}$. This derivative is the projection onto the tangent space T_xV of the derivative $Df(x) : \mathbb{R}^n \to \mathbb{R}^{n-m}$. Since this projection is equal to $I - DF(x)^\dagger DF(x)$ (see Definition 2.1) we obtain

$$D\hat{f}(x) = \pi_{T_xV} Df(x) = (I - DF(x)^\dagger DF(x))Df(x).$$

5 Experimental results.

Example 5.1 *Quadratic manifold.*

In this example we consider the quadratic manifold:

$$V = \{x \in \mathbb{R}^{100} \ : \sum_{k=1}^{5} \frac{x_k^2}{k} - \sum_{k=6}^{100} \frac{x_k^2}{k} = 1\}.$$

To compute the geodesics we use the Gauss method of order 4 with $\tau = 0.01$. On this manifold we solve the following problems:

1. A linear system $Bx = 0$, where B is a random 99×100 matrix,
2. A quadratic system: $Bx + c\|x\|^2 = 0$, where c is a given random vector in $\mathbb{R}^{99}$.

The initial point $x_0 \in \mathbb{R}^{100}$ of the Newton sequence is taken at random in following sense. Each component $x_{0k}, k = 1 \ldots 99$, is taken randomly in $[-1, 1]$ with respect to the uniform distribution and $x_{0,100}$ is computed to satisfy the equation defining V.

The corresponding results are displayed in the table 1 in the column "Geometric Newton Iteration" or "GNI".

For comparison we also display the results obtained for the same problems using the classical Newton method to the extended system

$$(F, f) : U \to \mathbb{R}^m \times \mathbb{R}^{n-m}$$

We call "Classical Newton Iteration" or "CNI" the corresponding sequence.

The typical behaviour of these iterations is shown in Fig. 1 and Table 1. They show a quadratic convergence obtained in few steps. Then we reach a limit due to round-off errors. The number of Newton steps is better for the geometric method than for the classical one but the precision is better for the classical method than for the geometric one. This is due to the amount of computation which is more important for the GNI.

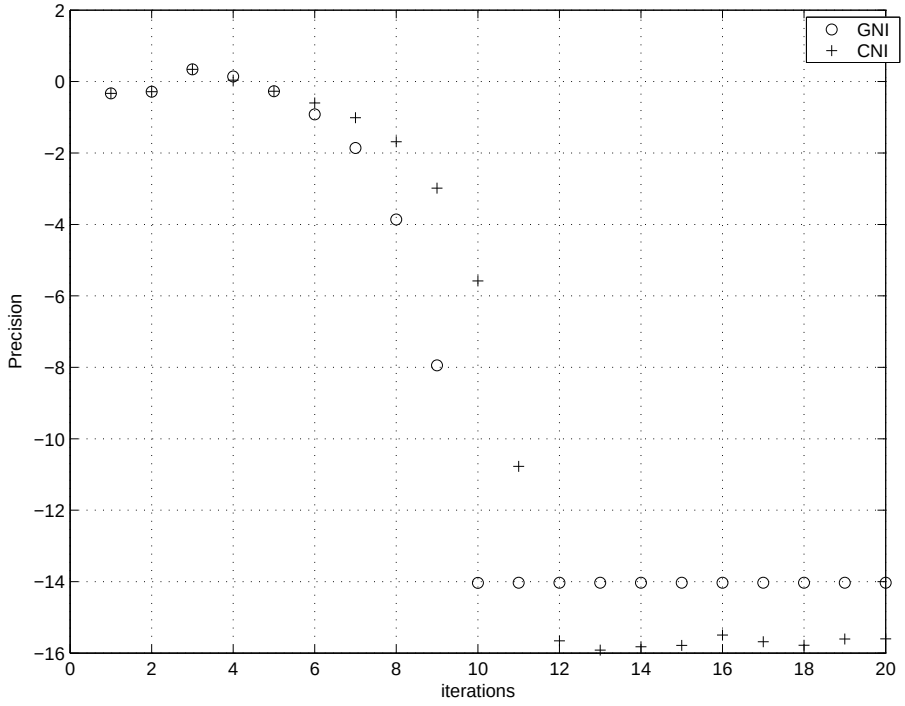


Figure 1: Convergence of geometric and classical Newton methods for the quadratic manifold. ∘ stands for GNI and + for CNI.

Problem	# of GNI steps	GNI Precision	# of CNI steps	CNI Precision
Linear	7	$1.01 \cdot 10^{-14}$	8	$2.2 \cdot 10^{-16}$
Quadratic	10	$1.06 \cdot 10^{-15}$	12	$3.0 \cdot 10^{-16}$

Table 1: Results for the quadratic manifold

Example 5.2 *"Distorted" quadratic manifolds.*

In this example we consider the following manifold:

$$V = \{x \in \mathbb{R}^{100} \ : \sum_{k=1}^{5} \frac{\sin(x_k)^2}{k} - \sum_{k=6}^{100} \frac{\sin(x_k)^2}{k} = 1\}.$$

On this manifold we solved the same test problems with the same parameters as in Example 1. This manifold has an infinite number of connected components. We restrict our study to the connected component V_0 such that $|x_i| \leq \pi/2$ for each $x \in V_0$. The initial point was taken randomly in V_0 like in the previous example. Under these conditions two solutions were found. The corresponding results are displayed in the table 2.

Problem	# of GNI steps	GNI Precision	# of CNI steps	CNI Precision
Linear	7	$7.1 \cdot 10^{-15}$	11	$1.4 \cdot 10^{-16}$
Quadratic	9	$2.1 \cdot 10^{-14}$	12	$2.2 \cdot 10^{-16}$

Table 2: Results for the distorted quadratic manifold.

Example 5.3 *Katsura's system.*

The following equations appear in a problem of magnetism in physics. For more details see Katsura-Sasaki [20], and also the web site [37].

$$\begin{aligned} & u_m = \sum_{i=-N}^{N} u_i u_{m-i} \ ; \ m = 0 \ldots N-1 \\ & \sum_{i=-N}^{N} u_i = 1 \\ & u_{-m} = u_m \ ; \ m = 1 \ldots 2N-1 \\ & u_m = 0; m = N+1 \ldots 2N-1 \end{aligned} \tag{16}$$

After eliminating u_m for $m \notin 0 \ldots N$ we obtain $N+1$ equations in $\mathbb{R}^{N+1}$:

$$\begin{aligned} & u_m = \sum_{i=m+1}^{N} u_i u_{i-m} + \sum_{i=0}^{N-m} u_i u_{i+m} + \sum_{i=1}^{m} u_i u_{m-i} \ ; \ m = 0 \ldots N-1 \\ & 2\sum_{i=1}^{N} u_i + u_0 = 1 \end{aligned} \tag{17}$$

This system is not a priori posed on a manifold. For this reason we split the equations into two groups: the M first equations from (17) (for $m = 0 \ldots M-1$) define a manifold V_M of codimension M, and the remaining equations are considered as a system on V_M. The GNI starts at a random point $x_0 \in V_M$. To find such a point we take at random a point y_0 in a box containing V_M (such a box is easy to compute from the structure of Katsura's system). Then we "project" y_0 on V via the Newton-Gauss method in $\mathbb{R}^{N+1}$.

In the next table we display the results for $N = 40$ and different values for M. We use the 4-order Gauss numerical integrator with $\tau = 0.01$. The results for the classical Newton method are also included: they correspond to the codimension $M = 0$. We don't know the number of real solutions of this system. During the test we found 4 different solutions.

In the following Figure we illustrate the same example with $N = 2$ and $M = 1$. The four first GNI iterates are located on the surface: $x_0, x_{1g}, x_{2g}, x_{3g}$ and x_{4g} while the iterates corresponding to the CNI ($x_0, x_{1c}, x_{2c}, x_{3c}, x_{4c}$) are clearly located outside the surface. We notice the same facts as in our first example: better numerical behaviour for the CNI but a better complexity in terms of the number of iterates for the GNI.

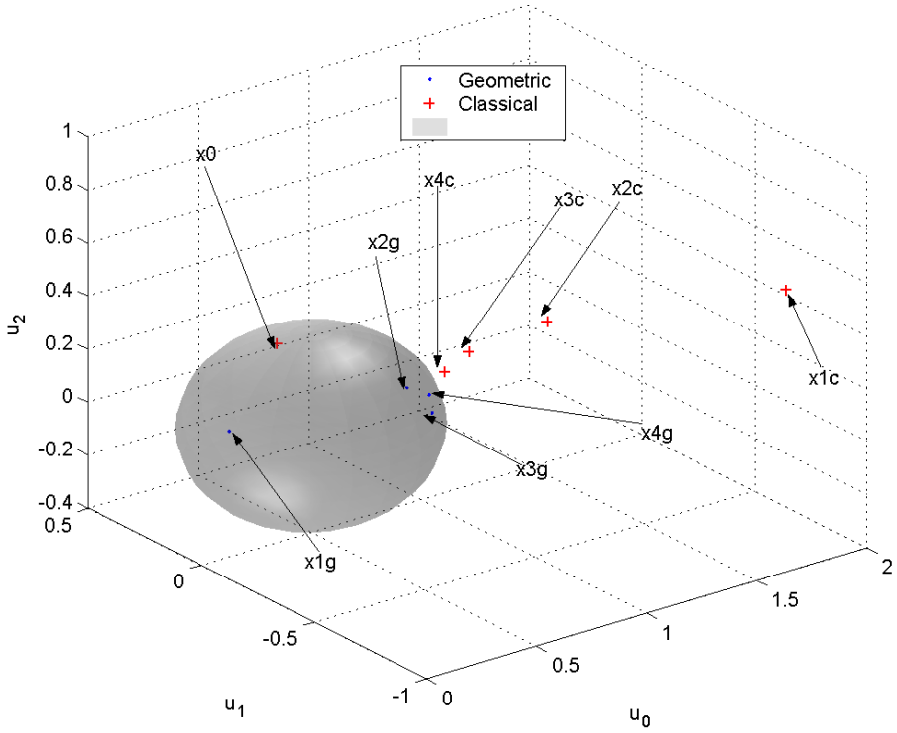


Figure 2: Convergence of CNI and GNI for the Katsura's example, N=2, M=1.

M	# of GNI steps	Precision
0 (CNI)	14	$1.78 \cdot 10^{-16}$
1	12	$4.4 \cdot 10^{-14}$
5	9	$6.7 \cdot 10^{-15}$
10	9	$5.6 \cdot 10^{-15}$
20	7	$1.0 \cdot 10^{-14}$
40	7	$3.2 \cdot 10^{-15}$

Table 3: Results for Katsura's example.

Example 5.4 *The generalised Brown system*

This example is a generalizaion of Brown's system, see Floudas-Pardolos [12] (Chapter 14, test problem 14.1.5):

$$\begin{array}{l} \sum\limits_{\substack{1 \le i \le N, \\ i \ne k}} x_i + 2x_k^r = N+1, \quad k = 1 \dots N-1 \\ \\ \prod\limits_{1 \le i \le N} x_i = 1 \end{array} \tag{18}$$

The case $N = 5$ and $r = 1$ corresponds to the original system.

Like in the previous example, this system is not a priori posed on a manifold. Its equations are also split into two groups in the same way.

The GNI starts at a random point x_0 in the manifold V of codimension M. To find such a point we take at random a point y_0 in the box $0.5 \le x_k \le 1.5$, $k = 1 \dots N$ with respect to the uniform distribution. Then, we project y_0 on V via the Newton-Gauss method in $\mathbb{R}^N$.

In the table 5 we display the results for $N = 10$, $r = 3$, and different values for M. We use the 4-order Gauss numerical integrator with $\tau = 0.01$. The results for the classical Newton method are also included: they correspond to the codimension $M = 0$. We don't know the number of real solution of this system. During the test we found the solution : $x = (1, \dots, 1)^T$.

M	# of GNI steps	Precision
0 (CNI)	8	$2.3 \cdot 10^{-15}$
1	7	$4.7 \cdot 10^{-14}$
3	6	$7.2 \cdot 10^{-14}$
5	6	$7.6 \cdot 10^{-14}$
9	6	$1.8 \cdot 10^{-14}$

Table 4: Results for generalized Brown example.

References

[1] Abraham R. and J. Marsden, *Foundations of Mechanics*. Addison-Wesley, 1978.

[2] Arnold V. I., *Mathematical Methods of Classical Mechanics*. Springer-Verlag NY, 1978.

[3] Adler R., J.-P. Dedieu, J. Margulies, M. Martens and M. Shub, *Newton Method on Riemannian Manifolds and a Geometric Model for the Human Spine*. IMA Journal of Numerical Analysis, 22 (2002) 1-32.

[4] Blum, L., F. Cucker, M. Shub and S. Smale, *Complexity and Real Computation*, Springer, 1998.

[5] Celledoni, E. and A. Iserles , *Methods for the approximation of the matrix exponential in a Lie-algebraic setting*, IMA Journal of Num. Anal. 21 (2001) 463-488.

[6] Dedieu, J.-P., G. Malajovich and P. Priouret, *Newton Method on Riemannian Manifolds: Covariant Alpha-Theory.*, to appear in IMA Journal of Numerical Analysis, 2003.

[7] Dedieu, J.-P. and M. Shub, *Multihomogeneous Newton's Method*. Mathematics of Computation, 69 (2000) 1071-1098.

[8] Dieudonné J., *Treatise on Analysis*, Academic Press, 1988.

[9] Do Carmo M., *Riemannian Geometry*, Birkhauser, Boston, 1992.

[10] Edelman, A., T. Arias and S. Smith, *The Geometry of Algorithms with Orthogonality Constraints*, SIAM J. Matrix Anal. Appl. 20 (1998) 303-353.

[11] Ferreira O., B. Svaiter, *Kantorovich's Theorem on Newton's Method in Riemannian Manifolds*. Journal of Complexity 18 (2002) 304-329.

[12] Floudas, C.A., Pardolos, P et al., *Handbook of test problems in local and global optimization*, Kluwer Academic Publishers, Dordrecht, 1999.

[13] Gallot S., D. Hulin, J. Lafontaine, *Riemannian Geometry*, Springer-Verlag, Berlin, 1993.

[14] Hairer, E., *Global modified Hamiltonian for constrained symplectic integrators.* Numer. Math. 95 (2003), no. 2, 325–336.

[15] Hairer, E., C. Lubich, *The life-span of backward error analysis for numerical integrators.* Numer. Math. 77 (1997), no. 2, 325–336.

[16] Hairer, E.; Norsett, S. P.; Wanner, G., *Solving ordinary differential equations. I. Nonstiff problems.* Second edition. Springer Series in Computational Mathematics, 8. Springer-Verlag, Berlin, 1993.

[17] Helgason, K., *Differential Geometry, Lie Groups, and Symmetric Spaces*, Academic Press, 1979.

[18] Iserles A., *On Cayley-transform methods for the discretization of Lie-group equations*, Foundations of Computational Mathematics (2001) vol. 1, pp. 129-160.

[19] Iserles A.,H.Z. Munthe-Kaas, S.P. Nørset, and A.Zanna, *Lie-group methods*, Acta Numerica (2001) vol. 9, pp. 215-365.

[20] Katsura S., M. Sasaki, *The asymmetric continuous distribution function of the effective field of the Ising model in the spin glass and the ferromagnetic states on the Bethe lattice*, Physica A, Vol. 157-3, (1989) pp. 1195-1202.

[21] Krogstad S., Munthe-Kaas H. and Zanna A., *Generalized Polar Coordinates on Lie Groups and Numerical Integrators*, Tech. Rep. No. 244, Dep. of Informatics, Univ. Bergen, (2003). To appear in Numerische Mathematik.

[22] Malajovich, G., *On Generalized Newton Algorithms*, Theoretical Computer Science, (1994), vol. 133, pp.65-84.

[23] Marsden, J. and T. Ratiu, *Introduction to Mechanics and Symmetry*, TAM 17, Springer-Verlag, 1994.

[24] Munthe-Kaas H. Z., Quispel G.R.W and Zanna A., *Generalized polar decompositions on Lie groups with involutive automorphisms*, Found. Comput. Math., 1 (2001), pp. 297–324.

[25] Noakes L., *A Global Algorithm for Geodesics*, J. Austral. Math. Soc. Ser. A, 64 (1998) 37-50.

[26] O'Neill B., *Semi-Riemannian Geometry*, Academic Press, New York, 1983.

[27] Owren, B. and B. Welfert, *The Newton Iteration on Lie Groups*, BIT, 40:1 (2000) 121-145.

[28] Sanz-Serna J. M., M. P. Calvo, *Numerical Hamiltonian Problems*, Chapman and Hall , 1994

[29] Shub, M., *Some Remarks on Dynamical Systems and Numerical Analysis*, in: Dynamical Systems and Partial Differential Equations, Proceedings of VII ELAM (L. Lara-Carrero and J. Lewowicz eds.), Equinoccio, Universidad Simon Bolivar, Caracas, 1986, 69-92.

[30] Shub, M., *Some Remarks on Bezout's Theorem and Complexity*, in *From Topology to Computation: Proceedings of the Smalefest*, J. M. Marsden, M. W. Hirsch and M. Shub eds., Springer, 1993, pp. 443-455.

[31] Shub, M., S. Smale, *Complexity of Bézout's Theorem I: Geometric Aspects*, J. Am. Math. Soc. (1993) 6 pp. 459-501.

[32] Shub, M., S. Smale, *Complexity of Bezout's Theorem IV: Probability of Success, Extensions*, SIAM J. Numer. Anal. (1996) vol. 33, pp. 128-148.

[33] Shub, M., S. Smale, *Complexity of Bézout's Theorem V: Polynomial Time*, Theoretical Computer Science, (1994) vol. 133, pp.141-164.

[34] Smith, S., *Optimization Techniques on Riemannian Manifolds*, in: Fields Institute Communications, vol. 3, AMS, 113-146, 1994.

[35] Udriste, C., *Convex Functions and Optimization Methods on Riemannian Manifolds*, Kluwer, 1994.

[36] Zanna A. and H. Z. Munthe-Kaas, *Generalized polar decompositions for the approximation of the matrix exponential*, Siam J. Matrix Anal. Appl., 23 (2001), pp. 840–862.

[37] http://www-sop.inria.fr/saga/POL/BASE/2.multipol/katsura.html

Chapter 2

Optimization, Riemannian Manifolds, Associative Memory, and Clustering

Dimitri Nowicki

Oleksiy Dekhtyarenko

[1]Institute for Mathematical Machines and Systems,

42, Glushkov ave., Kyiv 03187, Ukraine

Phone/fax: +380-44-2665587/2666457,
e-mail: nowicki@nnteam.org.ua , oleksiy@nnteam.org.ua

ABSTRACT: This paper is dedicated to the new algorithm for unsupervised learning and clustering. This algorithm is based on Hopfield-type pseudoinverse associative memory. We propose to represent synaptic matrices of this type of neural network as points on the Grassmann manifold. Then we establish the procedure of generalized averaging on this manifold. This procedure enables us to endow the associative memory with ability of data generalization. In the paper we provide experimental testing for the algorithm using simulated random data. After the synthesis of associative memory containing generalized data. Cluster centers are retrieved using procedure of associative recall with random starts.

1. Introduction

In this paper we apply geometric methods to neural associative memories. We use Riemannian manifolds arising from Linear algebra (like Stifel and Grassmann manifold) for representation of synaptic matrices of Hopfield-type neural networks. Using this approach we shall develop a neural algorithm for unsupervised learning and clustering.

Our algorithm is based on pseudoinverse associative memory [1]. Such a memory like other Hopfield-type networks is able to perform some kind of “unsupervised learning”: it can memorize unlabeled data. But such networks could not be used for clustering because they cannot generalize: training patterns are memorized “as is”. So, the network cannot retrieve

cluster centroids from large amount of data patterns.

In [2] and [3] there is developed a modification of projective associative memory that could do that. This algorithm possesses some properties of data generalization but weight matrix of the network is not projective. So, the network deteriorates as number of memorized data is augmented. Since certain number of training patterns the ability of associative recall is completely lost.

Unlike [2, 3] our method always produces projective matrices. Using techniques of generalized averaging over Riemannian manifold we construct the synaptic matrix of our network. Associative memory with such a matrix contains vectors generalizing training data. So, these vectors might be used as centroids of the clusters.
This method is related to averaging of subspaces [4], and optimization technique on the Grassmann manifold [5]. Applications of geometric methods to adaptive filtering are considered in [6]. Statistical estimation of invariant subspaces is investigated in [7] there Cramer-Rao bounds on Grassmann manifold are developed.

Since our method is based on non-iterative neural paradigm it has a good speed; only small number of epochs is needed even for large data sets. This feature makes associative-memory algorithm competitive in comparison with self-organizing maps (SOM) of Kohonen [8], the most known neural paradigm used for the purpose of clustering. Unfortunately training of SOMs is often very slow; millions of epochs are required for training of sufficiently-large network.

We provide experimental evidence for the associative-memory clustering. This method was tested using sufficiently large simulated data sets with intrinsic clustered structure.

2. Preliminaries

2.1. Projective associative memories

Our algorithms are based on associative memory with pseudoinverse learning rule [1]. This is a Hopfield-type auto-associative memory; memorized vectors are bipolar: $\mathbf{v}_k \in \{-1, 1\}^n$, $k=1\ldots m$. Suppose these vectors are columns of $n \times m$ matrix $\mathbf{V}$. Then synaptic matrix $\mathbf{C}$ of the memory is given by:

$$\mathbf{C}=\mathbf{V}\mathbf{V}^{+}, \qquad (1)$$

where $\mathbf{V}^{+}$ is a Moore-Penrose pseudoinverse or generalized inverse of $\mathbf{V}$. It might be computed directly as $\mathbf{V}^{+}=(\mathbf{V}^{\mathrm{T}}\mathbf{V})^{-1}\mathbf{V}^{\mathrm{T}}$ (for linearly independent columns of $\mathbf{V}$) or using Greville formulae (see, e.g., [9]) .

Associative recall is performed using following examination procedure: the input vector $\mathbf{x}_0$ is a starting point of the iterations:

$$\mathbf{x}_{t+1} = f(\mathbf{C}\mathbf{x}_t) \qquad (2)$$

where f is a monotonic odd function such that $\lim_{s\to\pm\infty} f(s) = \pm 1$ taken componentwise. The stable fixed point of this discrete-time dynamical system is called an attractor; the maximum Hamming distance between $\mathbf{x}_0$ and a memorized pattern $\mathbf{v}_k$ such that the examination procedure still converges to $\mathbf{v}_k$ is called an *attraction radius*.

We shall also use a *distinction coefficient* $r(\mathbf{x}, \mathbf{C})$ between a vector $\mathbf{x}$ and a projective matrix $\mathbf{C}$. Ii is given by:

$$r(\mathbf{C},\mathbf{x}) = \frac{\big((\mathbf{I}-\mathbf{C})\mathbf{x},\mathbf{x}\big)}{\|\mathbf{x}\|^2} = \left(\frac{\|(\mathbf{I}-\mathbf{C})\mathbf{x}\|}{\|\mathbf{x}\|}\right)^2) \qquad (3)$$

Note that $r(\mathbf{C},\mathbf{x})=0$ if $\mathbf{x}\in\mathrm{im}\mathbf{C}$ and $r(\mathbf{C},\mathbf{x})=1$ if $\mathbf{x}\in\ker\mathbf{C}$.

2.2 The Grassmann Manifold

The Grassmann manifold is a Riemannian manifold coming from linear algebra. This is the manifold of all m-dimensional subspaces in R^n, it is denoted by $G_{n,m}$. The Grassmann manifold could be defined as follows: at first we introduce the Stifiel manifold – a set of orthogonal $n\times m$-matrices $\mathbf{Y}$, $\mathbf{Y}^{\mathrm{T}}\mathbf{Y}=\mathbf{I}_{m\times m}$, endowed with Riemannian metric (this metric is induced by Euclidian norm in the space of $n\times m$-matrices). Then, we say that two matrices are equivalent if their columns span the same m-dimensional subspace. Equivalently, two matrices $\mathbf{Y}$ and $\mathbf{Y}'$ are equivalent if they are related by right multiplication of an orthogonal $m\times m$ matrix $\mathbf{U}$: $\mathbf{Y}'=\mathbf{Y}\mathbf{U}$.

Some computational algorithms on Grassmann manifold might be found in [5]. There are several representations of Grassmann manifold. We can represent elements of $G_{n,m}$ as (symmetric) $n\times n$ projection matrices of rank m. Indeed, there is one-to-one correspondence between such matrices and m-dimensional subspaces in R^n.

There are several ways to measure distance on $G_{n,m}$. Geodesic distance in Riemannian metric could be computed using SVD-decomposition (see [4]). One also can define a distance as a norm of difference between projective matrices. In this case the matrix 2-norm is usually taken. In order to reduce computational complexity of generalized averaging we use Frobenius matrix norm. So, the distance between projective matrices **X** and **Y** is

$$\rho(\mathbf{X},\mathbf{Y}) = \|\mathbf{X}-\mathbf{Y}\|_{Fro}$$

3. The Algorithm

3.1 Problem statement

Let us have a training sample containing K patterns $\mathbf{x}_1 \ldots \mathbf{x}_K \in \mathsf{R}^n$. Associative memory with generalized patterns is constructed as follows:

At first we create N groups of training vectors; each group contains m vectors. For each group data vectors are picked randomly from the training sample. The number $m<n$ should not exceed n; it is more or equal to desired quantity of clusters. Then we make N instances of pseudoinverse associative memory, each matrix stores one group of m training vectors. Synaptic matrices of these networks are $\mathbf{C}_k$, $k=1\ldots N$. To join all these instances of associative memory in one "generalized" network we use the procedure of generalized averaging.

3.2 Generalized averaging on the manifold

Consider a metric space M with metric $\rho(x,y)$ and a finite set $\{x_i\}_{i=1}^N \subset M$. The element

$$\bar{x} = \min_{x\in M} \sum_{i=1}^{N} \left(\rho(x,x_i)\right)^2 \quad (4)$$

is called the *generalized average* of points of this set. Similarly, the point

$$x_m = \min_{x\in M} \sum_{i=1}^{N} \rho(x,x_i) \quad (5)$$

is *a generalized median* of the same set. If M is an Euclidian space generalized average and median are usual average and median respectively. Generalized averaging is considered in [8],

problem of generalized averaging on homogenous manifolds might be found in [4].

3.3 Computing generalized average on the Grassmann manifold

Here we use representation of points of the Grassmann manifold $G_{n,m}$ as $n \times n$ (symmetric) projective matrices of rank m; the metric is induced by the Frobenius norm. Hence the problem of generalized average is equivalent to the following minimization problem:

$$\begin{aligned} &\min \varphi(\mathbf{X}) = \sum_{k=1}^{N} \left\| \mathbf{X} - \mathbf{C}_k \right\|^2 \\ &\mathbf{X}^2 = \mathbf{X}; \operatorname{rank} \mathbf{X} = m \end{aligned} \qquad (6)$$

Transform the objective function in the following way:

$$\varphi(\mathbf{X}) = \sum_{k=1}^{N} \sum_{i,j=1}^{n} (x_{ij} - c_{k,ij})^2 = \sum_{k=1}^{N} \sum_{i,j=1}^{n} \left(x_{ij}^2 - 2 x_{ij} c_{k,ij} + c_{k,ij}^2 \right) =$$

$$= \sum_{i,j=1}^{N} \left(N x_{ij}^2 - 2 x_{ij} \sum_{k=1}^{N} c_{ij,k} + \sum_{k=1}^{N} x_{ij,k}^2 \right) = \sum_{i,j=1}^{n} N \left(x_{ij} - \frac{1}{N} \sum_{k=1}^{N} c_{k,ij} \right)^2 + \text{const} =$$

$$= N \left\| \mathbf{X} - \frac{1}{N} \sum_{k=1}^{N} \mathbf{C}_k \right\|^2 + \text{const} = N \left\| \mathbf{X} - \overline{\mathbf{C}} \right\|^2 + \text{const}$$

Thus the problem (5) has been reduced to finding projective matrix of rank m closest to the simple average $\overline{\mathbf{C}}$ of the matrices $\mathbf{C}_k$.

Such a problem might be solved using Newton or conjugated-gradient methods on Grassmann manifold described in [5], [4] but for high-dimensional vectors this became computationally hard. In this paper we use a simplified approach.

3.4. Implementation

For tasks of neural networks there is need for computation with large matrices; typical dimension is several hundreds or thousands. So, many algorithms of constrained optimization or

optimization on Riemannian manifolds became inapplicable.

In order to simplify computation we construct a solution that is, in general, suboptimal. It is based on following suggestions:
Note that the Frobenius norm is invariant with respect to changing orthonormal basis. So, we can choose the basis of eigenvectors of $\overline{\mathbf{C}}$. Let them be ranged by way of decreasing of corresponding eigenvalues. In this basis $\overline{\mathbf{C}}$ is diagonal. We choose **X** equal to

$$\mathrm{diag}((\delta_k)_{k=1}^m) \qquad (7)$$

in this basis; where δ_k=1; k=1…m and δ_k=0 otherwise. Such a matrix is the closest to $\overline{\mathbf{C}}$ amongst projective matrices of rank m. Indeed, making non-diagonal elements non-zero just increases $\left\|\mathbf{X}-\overline{\mathbf{C}}\right\|^2$. Since **X** is diagonal (7) is the optimal solution. Thus **X** is a matrix of projection to the linear hull of m first eigenvectors of $\overline{\mathbf{C}}$.

3.5. Statistical Estimation

Now we introduce a model of random vector representing clustered data. We assume that this vector consists of finite number of centers (for each realization of this vector a center is taken randomly with certain probability) and additive noise. Under this assumption we can provide some estimation for the algorithm of associative clustering.

Proposition 3.1. Suppose the random vector **x** could be represented in the $\mathbf{x}=\mathbf{x}_0+\boldsymbol{\xi}$, where random vector $\mathbf{x}_0$ takes values $\mathbf{x}_0^{(1)},\dots\mathbf{x}_p^0$, and ξ is a random uncorrelated vector (with covariance matrix $\sigma^2\mathbf{I}$). Let matrix **C** be a projection matrix to the subspace spanned by $m>p$ vectors from this distribution. Then the invariant subspace of expectation of **C** is a subspace spanned by the centers.

Proof. We use orthogonal decomposition of **C**, $\mathbf{C}=\mathbf{Y}\mathbf{Y}^{\mathrm{T}}$: $\mathbf{Y}^{\mathrm{T}}\mathbf{Y}=\mathbf{I}$. The orthogonal matrix **Y** could be represented in the form $\mathbf{Y}=\mathbf{Y}^0+\boldsymbol{\eta}$, where columns of $\mathbf{Y}^0$ span the same subspace as $\mathbf{x}_0^{(1)},\dots\mathbf{x}_p^0$, and $\boldsymbol{\eta}$ is a zero-mean uncorrelated random matrix. One can compute the mean of **C**:

$$E\mathbf{C}=E(\mathbf{Y}_0+\boldsymbol{\eta})(\mathbf{Y}_0+\boldsymbol{\eta})^T=E\mathbf{Y}_0\mathbf{Y}_0{}^T+E\boldsymbol{\eta}\mathbf{Y}_0^Y+E\mathbf{Y}_0\boldsymbol{\eta}^T+E\boldsymbol{\eta}\boldsymbol{\eta}^T=$$
$$=\mathbf{Y}_0\mathbf{Y}_0^{T+}+m\mu\mathbf{I}=\mathbf{C}_0+m\mu\mathbf{I}$$

So, the matrix E**C** commutates with $\mathbf{C}_0$; therefore they have common invariant subspace. ■

The matrix computed according to section 3.4 from the sample average is close to the matrix $\mathbf{C}_0$ of projection to the subspace spanned by the cluster centers. Associative memory with such a synaptic matrix stores vectors which approximate unknown centers; these vectors could be extracted using the procedure of associative recall.

4. Experimental Technique

The goal of these series of experiments is to demonstrate network's ability to deal with data having predefined "clustered" structure. Training data could be divided into subsets grouping around the known centers. We are able to tell when the algorithm is able to retrieve these centers.

4.1. The Data

All experiments were carried out using 256-dimensional data vectors with bipolar component values {+1,-1}. The training set was generated as follows:

At first *p* cluster centers were produced; they were random bipolar vectors with equal probability of values. Then, data vectors themselves were constructed by adding a bipolar noise to center. More precisely, to make a data vector we took *h* randomly selected components of a center and changed their signs. Noise intensity *h* was random uniformly distributed number from 1 to *H*. We shall say that *H* is a *cluster radius*. For each cluster we generated equal number *N* of data points. We took *K*=1000 for all tests. Before entering to the network data were shuffled.

4.2. The Network

At first, *N* instances of associative memory were trained using pseudoinverse learning rule. Each network memorized *m* randomly picked data vectors. Synaptic matrices of these networks were averaged using the algorithm described above; and the resulting projective matrix **X** was obtained. The network with this matrix was used for simulations in order to retrieve cluster centers.

4.3. Finding Attractors

In order to find attractors we performed examination procedure (2) with activation function $f(x)=\text{sign}(x)$. Initial point was taken randomly; iterations were continued until a fixed point was reached.

Recall procedure ran T=10000 times; all attractors found were stored. Then the attractors were sorted by frequency or distinction coefficient.

5. Experimental results

In order to investigate network's behavior we performed experiments described above for different values of parameters. We used a network of 256 neurons and clusters with radius H=64. The matrix of the resulting network was computed by generalized averaging of N=1000 projective matrices. In these experiments all cluster centers were found by convergence from random starts. This was verified by comparing attractors found with centers; first p attractors were identical to centers.

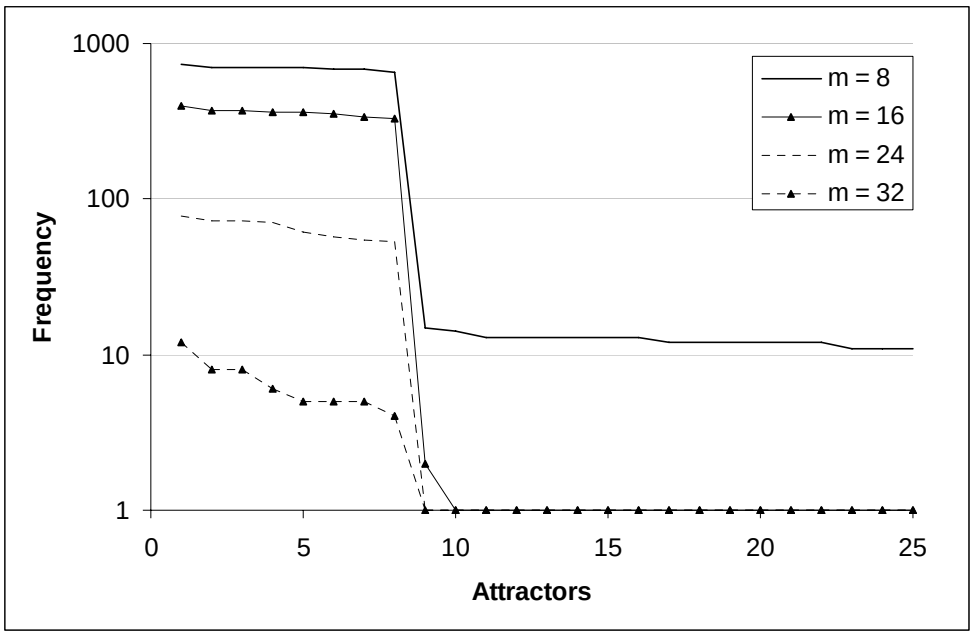


Fig. 1. Frequencies of attractors of associative clustering network for different m, p=8

Figure 1. corresponds to the case of constant number of clusters p=8; we varied invariant subspace dimension m. This parameter also means a number of data patterns stored in each instance of pseudoinverse associative memory. We can see that the algorithm works for large range of $m>p$. However, if m is large probability of convergence to a center decreases and

number of spurious attractors grows. For m=32 these probabilities have the same order; further increasing of m makes them identical; and centers will be lost.

The second series of experiments is related to the case of $m=p$. In the Fig. 2 attractors are sorted by frequency; difference between centers and spurious equilibria decreases as number of clusters grows. For $m=p$=32 the network was not able to solve its task; only 24 centers of 32 were found.

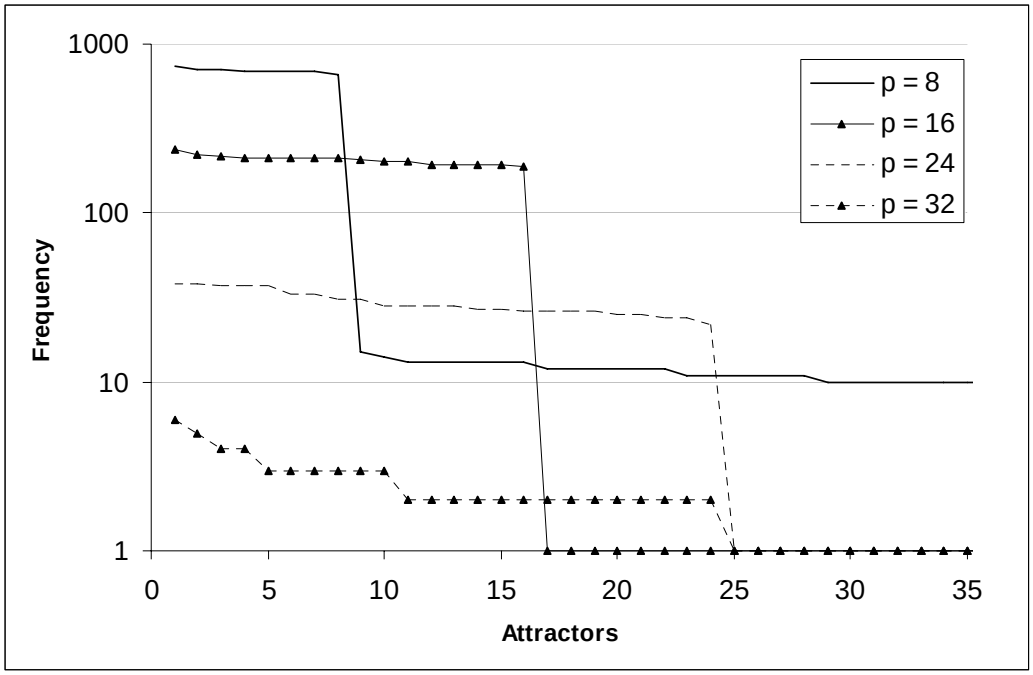


Fig. 2 Frequencies of attractors of associative clustering network for different p, and $m=p$

Figure 3 demonstrates another way of selecting attractors; here they are sorted by distinction coefficient $r(\mathbf{x},\mathbf{X})$ (3) with network's synaptic matrix. Results of this experiments show that difference of this measure between centers and spurious attractors is much stronger than for frequencies. This ratio is almost the same for different network configurations. So, the distinction coefficient might be used to reveal centers efficiently. Unfortunately, usage of this criterion combining with random starts cannot guarantee that number of network runs was sufficient to retrieve all centers. This can be seen from the results on fig. 3 for p = 32 – only 28 out of 32 centers were found using the value of distinction coefficient.

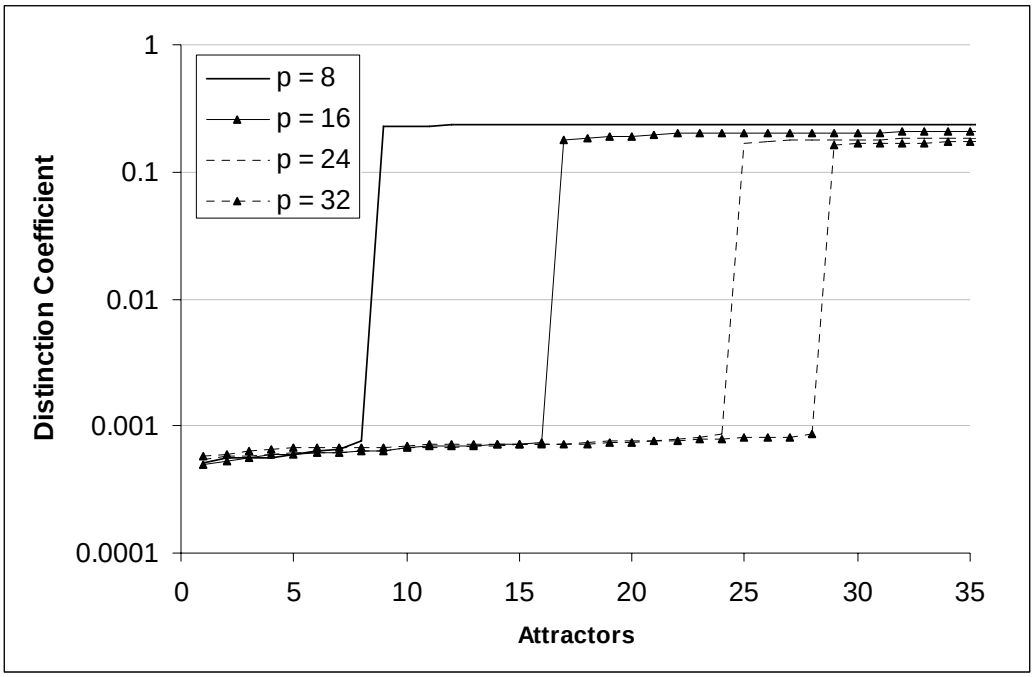


Fig. 3. Distinction coefficients of attractors of associative clustering network for different *p*, and *m*=*p*

Note also that usage of successive iterations (2) is necessary to find interesting attractors. If convergence to a stable state is not performed than the probabilities of finding cluster center or any spurious state are practically equal (especially for larger *m*).

6. The MNIST data

We also tested our method using the MNIST database of handwritten digits []. There are ten different classes of grayscale images (from ’0’ to ’9’, each of 28×28 pixels in size) together with their class labels. The training and test sets consist of 60,000 and 10,000 images respectively. The number of images in the training set varies from 5,842 to 6,742 per class and it is about 1,000 images per class in the test set. Each image was transformed into a *n*-dimensional vector ($n = 28 \times 28 = 784$). Then vectors were made bipolar using thresholding procedure.

.

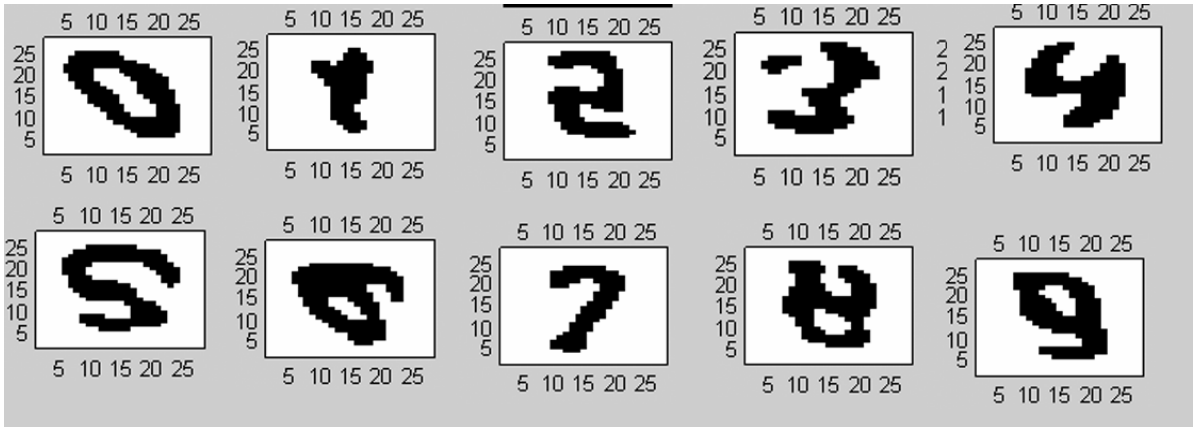


Fig. 4. Attractors of the associative memory obtained from MNIST data

In these experiment network's synaptic matrix was obtained as a generalized average of N=3750 matrices $\mathbf{C}_k$. Each of them contained m=16 randomly selected images. Such a network has many attractors; there are several attractors for each class. So, direct estimations of network's performance are difficult. Examples of attractors looking like 10 digits are displayed in fig. 3. These are generalized images of digits; they show that the network "knows" about the training data. Apart digit-like attractors the memory has some spurious ones looking like senseless shapes.

7. Conclusion

Experimental results described above show that proposed associative memories are able to generalize patterns. This makes them a good tool for clustering. Non-iterative nature of neural associative memories makes them quite attractive in comparison with many other neural algorithms of unsupervised learning.

Unfortunately, setting the value of parameter m in associative-memory clustering algorithms requires some a priori knowledge about data to be clustered. This value must be greater or equal than the number of clusters p, but, in the same time, must not exceed this number considerably. Moreover, m is bounded by the well known limitation on memory capacity of Hopfield-type NNs (which is of order n, preferably $m<0.3n$). This limitation might beeliminated by changing type of the manifold and/or metric used in of the main algorithm.

Note that this approach is based on optimization on Riemannian manifolds. This is a powerful technique that could be applied for some other tasks of learning and neural networks. In this

paper we used specific manifolds (Grassmann). For this manifold we selected only one type of distance (based on the Frobenius norm) and averaging. Moreover, the solution of corresponding optimization task was not exact. We expect that usage of different metric combining with exact geometric optimization may yield better performance of the associative-memory clustering. Development of appropriate techniques of high-dimensional optimization is a subject of the future work.

The proposed method may also be generalized for wider class of manifolds. In this case we should use geometric computation that works for arbitrary manifold (e.g. described in [10]). This extension of associative-clustering technique will enable to solve wider class of tasks.

References

1. L. Personnaz, I. Guyon, G. Dreyfus, "Collective computational properties of neural networks: New learning mechanisms," *Phys. Rev. A*, Vol.34 (5), pp. 4217-4228, 1986
2. Reznik A.M Non-Iterative Learning for Neural Networks. *Proceedings of the International Joint Conference on Neural Networks.* (Washington DC), July 10-16, 1999
3. A.S. Sitchov. *Methods of Improvement of Neural Associative Memory and its Application to Hybrid Modular Neural Networks.* Ph. D. thesis, IMMSP of NAS of Ukraine, Kyiv, Ukraine 2003 (in Ukrainian)
4. P.-A. Absil, R. Mahony, R. Sepulchre. Riemannian geometry of Grassmann manifolds with a view on algorithmic computation, *Acta Applicandae Mathematicae*, Vol. 80, No 2, pp. 199–220, Jan. 2004
5. A. Edelman, T. Arias, and S. Smith. The Geometry of Algorithms with Orthogonality Constraints. *Siam J. Matrix Anal. Appl.* Vol. 20, No. 2, pp. 303-353
6. S.T. Smith. *Geometric Optimization Methods for Adaptive Filtering.* Ph.D. Thesis . Harvard Univ. Cambridge MA, 1993
7. S.T. Smith. Intrinsic Cramer-Rao bounds and subspace estimation accuracy., 2004
8. Kohonen, Teuvo *Self-organizing maps.* Third edition. Springer Series in Information Sciences, 30. Springer-Verlag Berlin, 2001.
9. A. Albert, *Regression and the Moore-Penrose pseudoinverse,* Academic Press, New York-London, 1972
10. J.-P. Dedieu, D. Nowicki, Symplectic Methods for the Approximation of the Exponential Map and the Newton Iteration on Riemannian Submanifolds. Submitted to the *Journal of Complexity* (2004)

Chapter 3

Kernel-Based Associative Memory*

Dimitri Nowicki
Institute of Mathematical
Machines and Systems of NASU
Kiev, Ukraine
E-mail: nowicki@fromru.com

Oleksiy Dekhtyarenko
Institute of Mathematical
Machines and Systems of NASU
Kiev, Ukraine
E-mail: olexii@mail.ru

Abstract– **We propose a new approach to pseudo-inverse associative memories using kernel machine methodology. Basing on Hopfield-type pseudoinverse associative memories we developed a series of kernel-based hetero- and auto-associative algorithms. There are convergence processes possible during examination procedures even for continuous data. Kernel approach enables to overcome capacity limitations inherent to Hopfield-type networks. Memory capacity virtually does not depend on data dimension. We provide theoretical investigation for proposed methods and prove its attraction properties. Also we have experimentally tested them for tasks of classification and associative retrieval.**

I. Introduction

Nowadays there is a drastic growth in the domain of kernel machines. These methods and techniques are widely applied for pattern recognition, regression, classification and clustering. All kernel methods use feature space whose dimension is significantly larger than the dimension of original space. Original and feature spaces are connected by the nonlinear mapping:

$$\varphi : E_X \to E'_X .$$

Dimensionality of E'_X might be very large or even infinite. So it is difficult or impossible to operate with vectors in this space explicitly. However sometimes one needs to know only inner (scalar) product in E'_X as a function of elements of E_X. This function is called kernel:

$$K(u,v) = (\varphi(u), \varphi(v)) \qquad (1)$$

Most of kernel methods like SVM [1], LS-SVM [2] are used for classification, regression, and pattern recognition. Thus it will be interesting to construct kernel machines with other functions such as associative recall. The aim of this paper is to construct associative memory (AM) based on kernel machine.

Linear independence of memorized vectors is required by Hopfield-type associative memories. Moreover, their linear independence must be "sufficiently strong" i.e. every vector must be sufficiently far from the linear hull of all remaining ones. It implies that the number of patterns to memorize must be less than their dimension. In practice this number does not exceed 25% (pseudoinverse learning rule, [3]) or 70% of vectors' dimension (desaturation technique, see [4]). Diagonal elements of the synaptic matrix dominate if the number of patterns is close to this limit. This implies drastic decrease of attraction properties and deterioration of associative memory's capabilities.

Using kernel machines one can change over to the space where memorized data set becomes linearly independent. We use pseudoinverse hetero- and auto-associative memory as a prototype. Usage of kernel machines in scope of this paradigm enables to overcome limitations due to linearity of the basic model. In particular, we can remove capacity limitations of these memories. Using this approach we also constructed associative memory capable to iterative convergence during examination process with the continuous data.

II. The Algorithm

Lets consider pseudoinverse heteroassociative memory. Suppose E_X and E_X are input and output spaces with dimensionalities n and p respectively. We should store m pairs of vectors $x_i \in E_X, y_i \in E_Y, i = 1\text{K}\ m$. These vectors are supposed to form columns of matrices **X** and **Y** respectively. In order to provide appropriate heteroassociative behavior matrix B can be specified as:

$$\mathbf{BX} = \mathbf{Y}$$

whose solution is:

$$\mathbf{B} = \mathbf{Y}\mathbf{X}^{+} \qquad (2)$$

This matrix defines a projective operator $\mathbf{B} : E_X \to E_Y$ such that $\mathbf{B}\mathbf{x}_i = \mathbf{y}_i$ for all i. We denote by operator "$^+$" the Moore-Penrose pseudoinverse of **X** (see e.g. [5]). In case of linearly independent columns pseudoinverse matrix can be found as

$$\mathbf{X}^{+} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T = \mathbf{S}^{-1}\mathbf{X}^T. \qquad (3)$$

The elements of $m \times m$-sized matrix S are computed as pairwise scalar products of memorized vectors:

$$s_{ij} = (x_i, x_j). \qquad (4)$$

*This research was supported by INTAS grant #01-0257

Examination procedure takes an arbitrary input vector **x**. We should produce network's response **y**. This cold be done as follows:

$$\begin{aligned} y &= \mathbf{B}x = \mathbf{Y}\mathbf{S}^{-1}\mathbf{z}; \\ \mathbf{z} &= \mathbf{X}^T\mathbf{x}; \\ z_i &= (\mathbf{x}_i, \mathbf{x}). \end{aligned} \tag{5}$$

Note that we need to know only scalar products of memorized vectors themselves and input vector **x**.

We use this property of heteroassociative memory to construct the kernel algorithm. We replace E_X by E'_X whose dimensionality is $n' >> n$ (E'_X may also be an infinite-dimensional Hilbert space). Vectors in E'_X are evaluated using nonlinear transformation $\varphi : E_X \to E'_X$. E'_X is called *feature space*.

Let $x_i' = \varphi(x_i), x_i \in E_X$ be input vectors of training dataset, and $K(\mathbf{u}, \mathbf{v}) = (\varphi(\mathbf{u}), \varphi(\mathbf{v}))$ be a kernel.

Then, like (3-5) we get:

$$\begin{aligned} s_{ij} &= K(x_i, x_j); \\ z_i &= K(x_i, x). \end{aligned} \tag{6}$$

Expressions (2-3, 5-6) could be evaluated by means of kernel only, without explicit usage of E'_X, this leading to kernel-based procedures of learning and examination.

This is a basic algorithm for kernel associative memory.

Corollary 1 of Mercer's theorem: If $K(u,v)$ is a Mercer's kernel [1] then

1) Hilbert space E'_X and a mapping $\varphi : E_X \to E'_X$ exist such that $K(u, v) = (\varphi(u), \varphi(v))$
2) For each set of pairs $x_i \in E_X, y_i \in E_Y, i = 1\text{K } m$ matrix **S** is nonnegative-defined
3) If in addition $\dim(E_X') > m$, there exists a operator $B : E'_X \to E_Y$ such that $Bx'_i = y_i$ for all i.

Proof:

1) follows directly from Mercer's theorem
2) this is true because the matrix **S** consists of pairwise scalar products of $x_i' \in E'_X$ (it is a Gram matrix of this set of vectors)
3) such an operator could be built on the (m-dimensional) linear hull of $< \mathbf{x}_i' >_{i=1}^m \in E'_X$ and extended continuously to the whole E'_X. □

Mercer's condition is formulated as follows. Let $K(\mathbf{u}, \mathbf{v}) : Q \times Q \to \Re$ be a continuous symmetric function and Q be a compact set in E_X. Then, a space E'_X and a mapping $\varphi : E_X \to E'_X$ such that $\forall u, v \in Q \subset E_X \quad K(\mathbf{u}, \mathbf{v}) = \langle \varphi(\mathbf{u}), \varphi(\mathbf{v}) \rangle_{E'_X}$ exist if and only if for any $g \in L_2(Q)$ following inequality holds:

$$\iint_{u,v \in Q} K(u,v) g(u) g(v) \geq 0$$

Unfortunately, we cannot guarantee non-singularity of the matrix **S**. It is invertible if $< x_i' >_{i=1}^m \in E'_X$ are linearly independent. This condition may not hold for certain kernels and specific vector sets. In practice, one can suppress this problem using Tikhonov's regularization: instead of **S** using the matrix:

$$\mathbf{S}_\mu = \mathbf{S} + \mu \mathbf{I}$$

for small μ>0. This matrix is always invertible since **S** is nonnegative definite.

Another approach to this problem uses incremental construction of the matrix **S**. During each step of the algorithm its dimensionality is increased by one with the addition of each next memorized vector. If this leads to singular matrix, the vector is rejected. For inversion of **S** we use the technique for block matrices [5,6].

To memorize m patterns in this network we need to store $m \times m$-sized matrix **S**. We can say that kernel associative memory is capable to store as many images as neurons it has. This is a maximum estimation which is sometimes unreachable in practice. For instance, in case of *scalar-product* kernel this machine is identical to conventional neural heteroassociative memory.

III. Modifications of the Kernel Algorithm

A. Autoassociative memory

The algorithm described above might be also used for autoassociative memory. In this case E_x and E_y are identical, $x_i \in E_X, y_i \in E_Y, x_i = y_i, i = 1\text{K } m$. Matrix **S** is calculated by formula (6). There is an iterative examination procedure: the vector x_t is sent to the network's input, using (5-6) we obtain postsynaptic potential y_t. Then, in case of bipolar data we apply activation function and compute the next state of the system:

$$x_{t+1} = f(y_t) \tag{7}$$

This procedure is iterated until a stable state (attractor) has been reached. Attractors of such systems are described by

Theorem 1. *Suppose for autoassociative memory (4-8) conditions of the corollary 1 hold, and matrix* **S** *is invertible. Then attractors of corresponding examination procedure are only fixed points or 2-cycles*

Proof: We construct energy function in the way similar to the corresponding proof for Hopfield networks:

$$E_t = -\frac{1}{2}K(x_t, y_t) \tag{8}$$

By corollary from Mercer's theorem a self-conjugated operator $C: E'_x \to E_y$ exists such that $Cx'_t = y_t$. Applying properties of scalar product in E'_x we get:

$$\begin{aligned} E_t - E_{t+1} &= -\frac{1}{2}(x'_t, Cx'_{t-1}) + \frac{1}{2}(x'_{t+1}, Cx'_t) = \\ -\frac{1}{2}(x'_{t-1}, Cx'_t) &+ \frac{1}{2}(x'_{t+1}, Cx'_t) = \\ &= \frac{1}{2}K(y_t, x_{t+1}) - \frac{1}{2}K(y_t, x_{t-1}) \end{aligned} \tag{9}$$

Since the kernel is monotonic function with respect to distance between x and y expression (9) is non-negative, it is zero if and only if the fixed point is reached.□

The scheme of examination algorithm for auto-associative memory is displayed in the fig. 1.

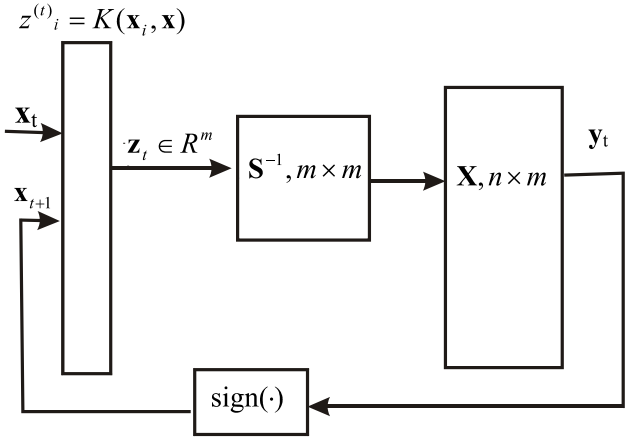


Fig. 1. Scheme of the kernel autoassociative memory

B. Internal activation function

Consider the vector $\mathbf{w} = \mathbf{S}^{-1}\mathbf{z}$. It corresponds exactly to *k*-th memorized pattern if and only if $w_i = \delta_{ik}$. To provide a better convergence to such *w* we apply *internal activation function*:

$$F: \mathbf{w} \to \mathbf{w}' \; w' = \theta(w_i)$$

where $\theta: [0,1] \to [0,1]$ is smooth monotonic function such that $\theta(0) = 0, \theta(1) = 1; \theta'(0) = \theta'(1) = 0$.

IV. Experimental results

Our models and algorithms were experimentally tested for several tasks of auto- and heteroassociative recall and classification. Here we display results or auto-associative memory working with simulated data arrays and real-world data (images). For experiments we used following three types of kernel:

1. Polynomial
$$K(x,y) = (1 + \alpha(x,y))^{\beta}, \quad \alpha > 0,\ \beta - \text{positive integer} \tag{10}$$
2. Gaussian RBF
$$K(x,y) = \exp\left(-\alpha \|x - y\|^2\right), \alpha > 0 \tag{11}$$
3. Power RBF:
$$K(x,y) = \left(1 + \alpha \|x - y\|^2\right)^{\beta}, \quad \alpha > 0, \beta > 0 \tag{12}$$

We studied attraction properties of kernel associative memory. All experiments were performed using internal activation function and iterative examination procedure. Algorithms were implemented using neural-network software package NeuroLand [7].

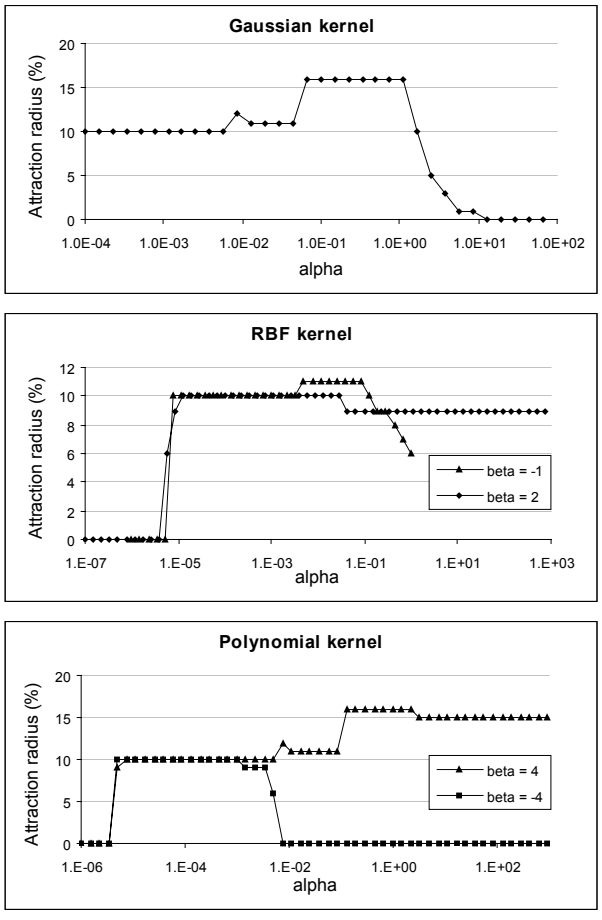


Fig. 2. Attraction radius of kernel AM for bipolar data

A. Simulated bipolar data

To study attraction properties of kernel associative memory for bipolar data we choose a network memorizing 264 64-dimensional patterns. Bipolar data

vectors were randomly generated, probabilities of values +1 and -1 for each component were equal, and components were independent.

Attraction radius was measured as a maximum value of bipolar noise such that the network still gave correct responses for all memorized patterns. In fig. 2 we display attraction radius depending on parameters for kernels (10-12).

B. Image data

In these experiments we used 30×30 gray-scale photographs of faces. They were presented as real-valued vectors with components normalized to [-1;1], the kernel AM memorized 61 images.

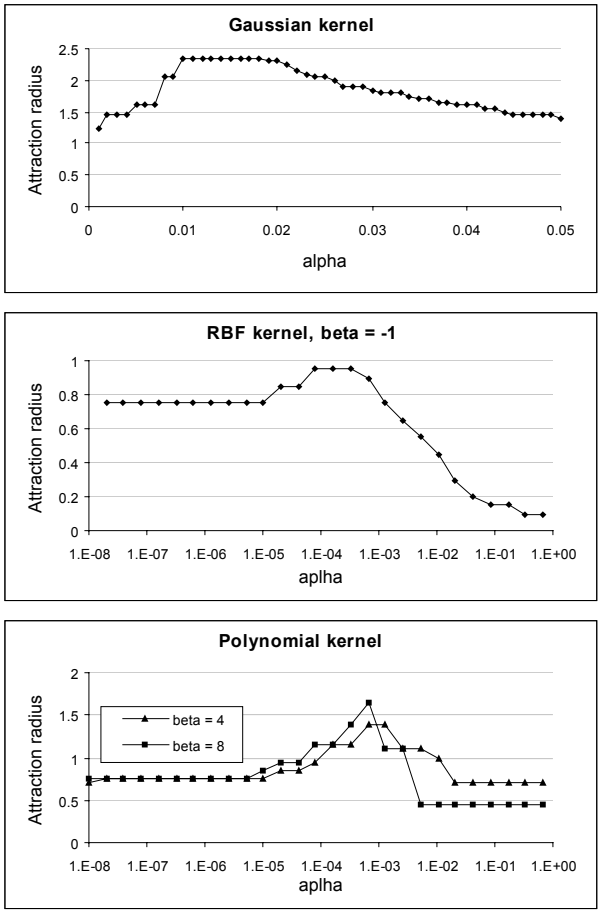


Fig. 3. Attraction radius of kernel AM for picture data

Input vectors were obtained from original patterns by adding Gaussian noise with zero mean. Standard deviation σ of this noise served as a measure of attraction radius. More precisely, attraction radius was set equal to σ such that all images were restored with fixed precision ε. Attraction radius depending on parameters for kernels (10-12) is displayed in fig. 3.

V. Conclusion

This article introduces associative memory based on kernel machine. We present theoretical justification and experimental tests for these techniques.

Unlike [8], where author uses high order generalization of the Hopfield model that includes interactions between more than two neurons, we restrict ourselves to two component Hamiltonian (energy function). Doing so we are able to provide analytical solution for the stability equation (2).

Experimental results show that proposed kernel algorithm successfully works as auto- and heteroassociative memory. We demonstrate attraction properties of kernel AM for different types of data. It may also be used for classification and pattern recognition. Using kernel methods we can construct iterative examination procedure even for continuous data. Also we can increase capacity of associative memory and overcome limitations inherent to Hopfield-type neural networks.

References

[1] V. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, NY, 1998.

[2] Smale S. On the Mathematical Foundations of Learning *Bull. Am. Math. Soc*., Vol. 39, No. 1, pp. 1-49, 2001.

[3] L. Personnaz, I. Guyon, G. Dreyfus, "Collective computational properties of neural networks: New learning mechanisms," *Phys. Rev. A*, Vol.34 (5), pp. 4217-4228, 1986.

[4] D.O. Gorodnichy, A.M. Reznik, "Increasing Attraction of Pseudo-Inverse Neural Networks," *Neural Processing Letters,* vol. 5, pp. 121-125, 1997.

[5] A. Albert, *Regression and the Moore-Penrose pseudoinverse,* Academic Press, New York-London, 1972.

[6] L.A. Pipes, *Applied mathematics for engineers and physicists*, McGraw-Hill Book Co., New York-Toronto-London, 1958.

[7] A.M. Reznik, E.A. Kalina, A.S. Sitchov, E.G. Sadovaya, O.K. Dekhtyarenko, A.A. Galinskaya, "The multifunctional neural computer NeuroLand," *Proceedings of the Int. Conf. on Inductive Simulation,* Lviv, Ukraine, vol.1 (4), pp. 82-88. May 20-25, 2002.

[8] Barbara Caputo, "Storage Capacity of Kernel Associative Memories," *Proceedings of the Int. Conf. on Art. Neural Networks,* Aug. 27-31 2002, Madrid, Spain.

Associative Memories with "Killed" Neurons: the Methods of Recovery*

A.M. Reznik, A.S. Sitchov, O.K. Dekhtyarenko, and D.W. Nowicki
The Institute of the Mathematical Machines and Systems, Ukrainian National Academy of Science
03187, 42 Glushkov Str, Kiev, Ukraine
neuro@immsp.kiev.ua

***Abstract*–We consider re-learning ability of a Hopfield-type network after killing some neurons. Neurons were "killed" by means of nullification of corresponding rows and columns of the synaptic matrix. We show that one can restore recognition ability of this network using re-training with the vectors, which were memorized before. The number of vectors needed is equal to the number of deleted neurons. It does not depend on network's size and on volume of stored data.**

INTRODUCTION

It is known that a classical Hopfield network has decreasing convergence ability with respect to number of memorized vectors. Such a network cannot store more vectors than *14*% of neurons' number [1]. Pseudoinverse learning rule enables to increase this ratio up to 25% [2]. In this case we must to store exact weight values in the synaptic matrix (at least *7* bits per weight, [3]). But sometimes disturbance of accurate weight values does not decreases convergence ability of the Hopfield-type network. On the contrary, some distortions may make it work better as an associative memory. Let us note some examples of such "useful distortions": methods of desaturation [4], which allows increasing of the memorize ability about 5 times, pseudoinverse adaptive filter [5], method of weight selection (it enables to reduce number of weights to *30*% of original quantity not worsening associative-memory capabilities, [6]). These examples illustrate the effect of information redundancy inherent to Hopfield-type associative memories

Therefore the following question seems to be interesting: Does the redundancy effect work if some neurons of the Hopfield-type network are completely destroyed? Could one recover the associative memory in this case? To answer these questions we consider a pseudo-inverse network with some neurons "killed" by means of nullification of all their synaptic weights (both for the inputs and the outputs). Once exposed to such a distortion, the network loses its ability to converge, i.e. the destruction of associative memory takes place and all its content becomes inaccessible. We show that it is possible to recover the network completely via retraining it with some of the previously stored vectors. The number of vectors needed is equal to the number of deleted neurons. It does not depend on network's size and on volume of stored data. This phenomenon looks like recovery of amnesia patients after reminding them significant events of their past.

*This research was supported by INTAS-01-0257

THE MODEL OF NEURAL NETWORK

According to the J. Hopfield's scheme each neuron is connected to each other and itself; forward and backward connections have the same weight. Weighted sum of output signals forms a postsynaptic potential (PSP). Depending on a sign of this sum the neuron's output possess the values *+1* or *-1*. The outputs form a *N*-dimensional vector of current network's state, and weights form a synaptic matrix *C*. Under certain conditions for this matrix the network has stable states called attractors. If a network state isn't an attractor the process of convergence will take place. Convergence ends in a nearest attractor. The convergence process looks like an associative recall, that's why networks of this type are known as associative memories. The algorithm for calculation of synaptic matrix from given set of attractors [2] is based on solution of the stable state equation:

$$CU = U. \tag{1}$$

Here U is a matrix $M \times N$. The vectors of desired attractor states are its columns. The solution of this equation has a form:

$$C = UU^{+}, \tag{2}$$

Besides the main attractors defined by equation (1) the network has spurious equilibria defined as solutions of non-linear stability equation.

If there are many spurious attractors the network may stop at them before converging to main attractors. Spurious attractors exert influence only if $M/N > 1/10$. For slightly saturated networks they could be neglected. The pseudoinverse algorithm is commonly used to compute value of the matrix C. It enables to successively update the matrix in memorizing of each vector from U [4].

$$C^m = C^{m-1} + \Delta^m;$$
$$\Delta^M = (I - C^{m-1})UU^T(I - C^{m-1})/\, U^T(I - C^{m-1})U, \tag{3}$$

where C^m is a value of synaptic matrix after memorizing M vectors.

An expression for Δ^m could be rewritten in the form:

$$\Delta_{ij}^{M} = (u_i - s_i)\,(u_j - s_j)/q, \tag{4}$$

where $s_i = \sum_{j=1}^{N} C_{ij} u_j$ is a postsynaptic potential of j-th neuron; $q = \sum_{j=1}^{N} (u_j - s_j) u_j$

A normed value of q is called coefficient of distinction. It describes a component of vector U orthogonal to *m-1* previously stored vectors.

$$k = q / N. \quad (5)$$

The value of k decreases as U approaches to linear hull of U_i and reaches zero when U belongs to this linear hull. Such a behavior resembles the resonance effect as it emphasizes the difference between similar stored vectors.

Quality of associative memories could be also described in terms of attraction radius. Direct attraction radius is the largest value of Hamming distance between initial point and the attractor for that the examination procedure starting at this point will still converge to the corresponding attractor during one iteration.

The value of direct attraction radius depends on data nature and could be estimated by the following expression [4]:

$$H < 0.5(N-1)^{0,5}[1-(1+\alpha)M/N][M/N-(M/N)^2]^{-0,5}+1 \quad (6)$$

A positive value $\alpha<1$ is called desaturation coefficient. Diagonal terms of the matrix should be multiplied by this value to fasten convergence and to resist saturation effects.

During experimentation we use the concept of the full attraction radius, which is the maximum Hamming distance covered by the network as it reaches the state of attraction (regardless of the number of iterations). The full attraction radius normally is 3-5 times greater than attraction radius given by (6) and is more important for the practical applications.

Change of Attraction Properties after Removing Connections

Partial removal of connections causes change of the PSP. Small changes do not affect its sign; so they don't cause network's state changes. In this case the effect could be seen due to change of the distinction coefficient. This coefficient describes network's behavior in the neighborhood of the attractor.

The histograms of k are for the network of *256* neurons displayed in the fig. 1. In this network all connections of n (randomly selected) neurons were destroyed after memorizing M vectors. For these experiments the NeuroLand software package [6] was used. Memorized and test vectors were generated randomly. We can see that the distribution of k looks like normal. The removal of just 10% of neurons leads to the significant non-zero k values that signifies the loss of the synaptic matrix projective properties and the deterioration of the network recall capacities.

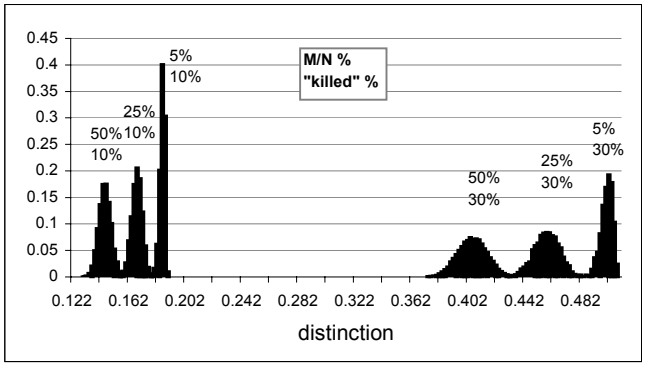


Fig. 1. The k value distinction for various memory saturations (M/N) and "killed" neuron portions.

Training Distorted Neural Network

In the fig. 1 we can see resonance properties of the NN changed. These changes increase a probability of stop in the spurious attractors. So the volume of data recalled is decreased. Moreover, loss of projectivity of the synaptic matrix may damage memorizing recall abilities more seriously. If one uses formulae (3, 4) to correct perturbed matrix then distortion will be only cumulated. That does finally completely destroy the associative memory. To verify these assumptions we have made some experiments. Under certain conditions the results were completely opposite.

The results for memorizing *120* vectors and "killing" *40* neurons in the network of *256* neurons are presented in the figs. 2 and 3. For examination we used desaturation with $\alpha = 0.1$. According to formula (6) $H = 8.7$, and the complete attraction radius was about *35*. To obtain the AR experimentally we use noisy values of stored vectors as initial conditions for the convergence process. Complete attraction radius was defined as a maximum value of H for that network was converged to the correct image for *95* of *100* noise instances.

Data for "killing" *40* neurons and re-memorizing *10*, *30*, and *40* vectors are displayed in the fig. 2, 3. Vectors for retrain were randomly selected from images initially stored. Fig 2 depicts the values of attraction radius for the remaining part of the network (*216* neurons).

We can see that the destroyed network completely losses its convergence ability. Only for one of *120* initially stored images (*#100*) attraction radius was non-zero. During additional training attraction radius is recovered completely for retrain images and partially fore the rest. As number of retrain vectors gets equal to number of "killed" neurons attraction properties of the network are completely restored. The distinction coefficient with respect to retrain dynamics is shown the figure 3. Unlike fig. 2 this graph takes into account all *256* neurons of the network. Dynamics of k with respect to retrain looks like attraction radius dependencies. After re-memorizing *40* vectors it turns to zero for all stored images.

The recovery effect for associative memories takes place if

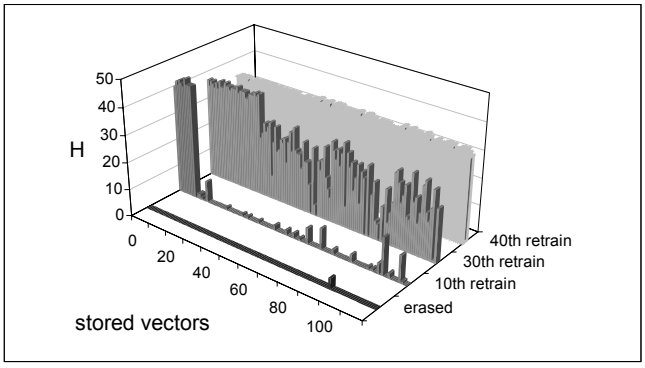


Fig. 2. The changes of attraction radius with net retrain.

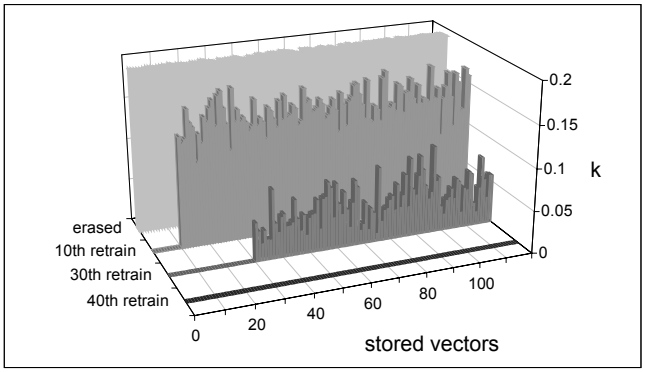


Fig. 3. The changes of distinction coefficient with net retrain

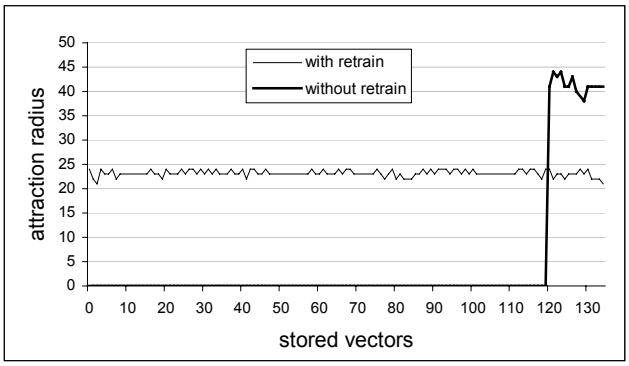


Fig. 4. The change of attraction radius with net extra train

and only if retrain data belongs to a set of initial images. If we take different vectors attraction abilities of the network could not be restored. That follows from fig. 4: attraction radius after storing *15* new vectors in the network. These are vectors *#121* and more. The dependencies for new training that is done immediately after killing *40* neurons (bold line) or following retraining (thin line) are displayed in this graph. We can see that without retraining attraction radius is zero for all previously stored vectors. For the new stored vectors attraction radius (it was more than *40*) increases in comparison with the value before damaging network. It could be explained by the fact that the trace of the synaptic matrix was decreased by the neurons removal, thus the sum of eigenvalues defining the associative memory saturation became smaller.

SPECTRA OF DISTORTED SYNAPTIC MATRICES

The recovery process of associative memory could be observed by examining spectra of synaptic matrices. Data for the matrix *256×256* with *120* stored vectors after nullification of *40* rows and columns are displayed in fig 5. There are graphs of sorted spectra before retraining and after *10*, *30*, and *40* retrain vectors. We can see that "killing" neurons preserves rank of the matrix, although values of *40* eigenvalues of *120* strongly decreased. After storing each following vector rank of the matrix increases by one, an eigenvector with value of *1* is added, *40* weakened eigenvalues are changing. After retrain they form a tail of the spectrum; these eigenvalues are less than *0.47*. There is an orthogonal component with respect to original memorized vectors in eigenvectors of the corrupted matrix.

Fig. 5 displays data for dense filling of the memory (*M/N>=0.47*) after removal of *15%* of neurons. Fig. 6 displays data for slight filling of the same network (*M/N>=0.15*) after removal of *10* neurons only. There are spectra before retraining, after *5* and *10* retrain vectors and after additional training with 20 new vectors.

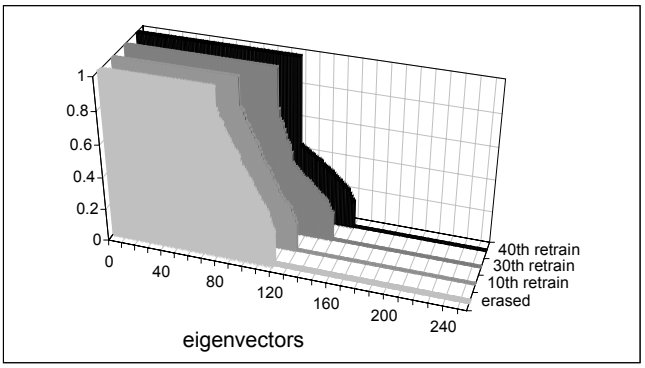


Fig. 5. Spectra evolution during the net retrain (*M = 120*)

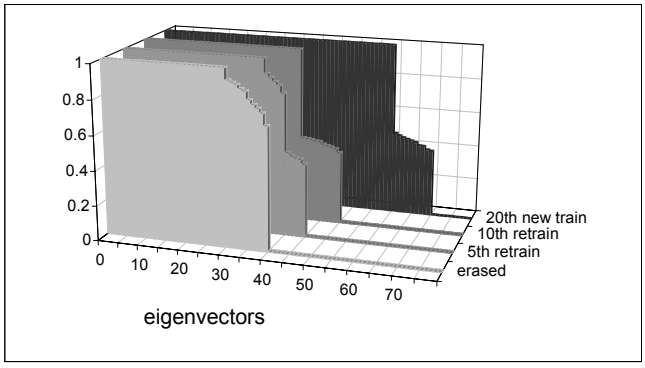


Fig. 6. Spectra evolution during net retrain and extra train with new data (*M = 40+20*)

Note that *20* new eigenvectors with $\lambda = 1$ almost do not influence on *10* small eigenvalues (appeared after corruption).

For slightly saturated memory non-projectivity of the matrix does not practically influence on the network's behavior. However for strongly filled memory there are many spurious attractors; so, available associative memory is reduced. We can depress spurious attractors using desaturation. To reduce non-projectivity more drastically one could raise the synaptic matrix to high power; so only eigenvalues close to *1* would be preserved.

CONCLUSION

Considering the results of our experiments it is worth to emphasize that the discussed phenomenon covers not only a simple substitution of "killed" neurons and their connections, but also the complete recovery of the autoassociative memory functions for the remaining part of the net (its functionality was lost due to the loss of the network ability to converge towards attractors). Therefore it is possible to say about the recovery both of stored data and the autoassociative access mechanism.

We have done experiments on network recovery for Hopfield-type networks with *64* to *2048* neurons for different levels of saturation and corruption. In all the case the network could be completely recovered using retraining with original data.

The number of retrained images needed for the complete recovery was always equal to the number of deleted neurons. We have not revealed any differences caused by the various possible selections of retrained images from the already stored ones in the network. Such independence on the retrain set selection could be used for designing of ultra stable systems that are capable to preserve their features in spite of constant resource degradation. In order to achieve it, these systems must undergo recovery via retraining of a control dataset more intensively than the resource degradation occurs.

This phenomenon looks like recovery of amnesia patients after reminding them significant events of their past. So, we can make the conjecture that in the biological neural systems there are mechanisms working like pseudo-inverse associative memories. We can cite some facts of biological neuroscience to support this conjecture. Not that most nerve fibers are less than *2* mm long; size of the dendrite branching area (*0.1-0.4* mm) has the same order. In *1* mm^2 there are thousands strongly connected neurons. So, they form local structures of *100-1000* neurons looking like Hopfield-type networks. We can suppose that these structures perform information memorizing and retrieval. In scope of our model and results long-term memory acts by systematic reactivation of local structures; they are re-training with previously memorized patterns. Memory corruption is prevented due to continuous conscious and subconscious work of the brain; also due to the dream activity. The further research in cooperation with specialists in biological neuroscience and psychologists is needed to verify these conjectures.

REFERENCES

[1] J.J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *in Proc. Nat. Acad. Sci.,* vol. 79, pp. 2554-2558, Apr. 1982.

[2] L. Personnaz, I. Guyon, G. Dreyfus, "Collective computational properties of neural networks: New learning mechanisms," *Phys. Rev. A.,* vol. 34, no. 5, pp. 4217-4228, 1986.

[3] M. Weinfield, "A fully digital integrated CMOS Hopfield network including learning algorithm," *in Proc. Int. Workshop WLSI Art. Intell.,* Univ. of Oxford, E1-E10, 1988.

[4] A.M. Reznik, D.O. Gorodnichy, A.S. Sitchov, "Regulating feedback bond in neural networks with learning projectional algorithm," *Cybernetics and system analysis*, vol. 32, no. 6, pp. 868-875, 1996.

[5] A.M Reznik, "Non-iterative learning for neural networks," *in Proc. Int. Joint Conf. Neural Networks,* Washington, no. 548, July, 1999.

[6] A.S. Sitchov, "Weight selection in neural networks with pseudoinverse learning rule," (in Russian) *Mathematical Machines and Systems,* vol.2, pp. 25-30, 1998.

[7] A. M. Reznik, E.A. Kalina, A.S. Sitchov, E.G. Sadovaya, O.K. Dekhtyarenko, A.A. Galinskaya, "Multifunctional neurocomputer NeuroLand," (in Russian) *in Proc. Int. Conf. Inductive Simulation,* Lviv, Ukraine, vol. 1(4), pp. 82-88, May 2002.

Chapter 4

Filtering of Deep Brain Stimulation Artifacts out Using a Nonlinear Oscillatory Model

Aksenova T.I.[1,2], Nowicki D.V.[*1,3], Benabid A-L[1]
[1] Unit 318, INSERM
CHU A. Michallon, BP 217, 38043 Grenoble, Cedex 09, France

[2] Institute of Applied System Analysis, Ukrainian Academy of Sciences,
Prospekt Peremogy, 37, Kiev 03056, Ukraine
[3]Institute of Mathematical Machines and Systems
42 Glushkov ave., 03187 Kiev Ukraine

E-mail:
tatyana.aksyonova@ujf-grenoble.fr
nowicki@nnteam.org.ua
Alim-Louis.Benabid@ujf-grenoble.fr

ABSTRACT

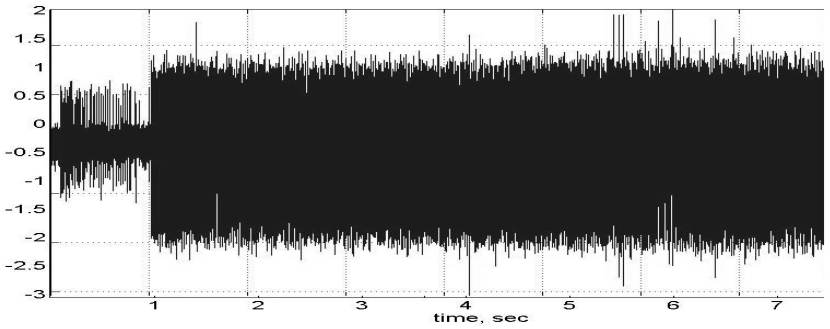
The present paper is devoted to suppression of spurious signals (artifacts) in records of neural activity during deep brain stimulation.. An approach based on nonlinear adaptive model with self-oscillations is proposed. We developed an algorithm of adaptative filtering basing on this approach. The proposed algorithm was tested using records collected from patients during the stimulation. Then it was compared to existing methods, and showed the best performance.

1. Introduction

This paper presents an approach for filtering signals of neuronal activity during Deep Brain Stimulation (DBS) using nonlinear oscillatory models. High-frequency (100-300 Hz) DBS is a surgical procedure for treating a variety of disabling neurological symptoms, in particular due to Parkinson's disease. In spite of its clinical efficiency over 20 years the mechanism of action of DBS is still a matter of debate (Benabid et al, 2005; McIntyre CC et all, 2004). Understanding how DBS at high frequency works is of paramount importance as this will provide the understanding of the circuitry of basal ganglia. The major difficulty of to study the mechanism of action of DBS is that the appropriate signal of neuronal activity during the stimulation, namely the extracellular microelectrode recording of action potentials (spikes), cannot be analyzed directly due to stimulation artifacts present in the records (Fig. 1a, 2a). The artifacts are induced by the periodically repeated electrical impulses delivered to the target zone in the brain. The artifacts have a common waveform but are not identical due sampling errors and irregularities of stimulus production. The Artifact-to-Signal Ratio (ASR), which is the ratio of the mean of amplitudes of artifacts to the averaged amplitude of spikes of neuronal activity observed before the stimulation, varies between 5 to 20. Typical artifact duration in the timeline is 20-50% of each stimulation period (Fig. 3a) although the pulse width of the stimulating pulse (60μs) is less than 1%

[*] D.V. Nowicki is a corresponding author

a)



b)

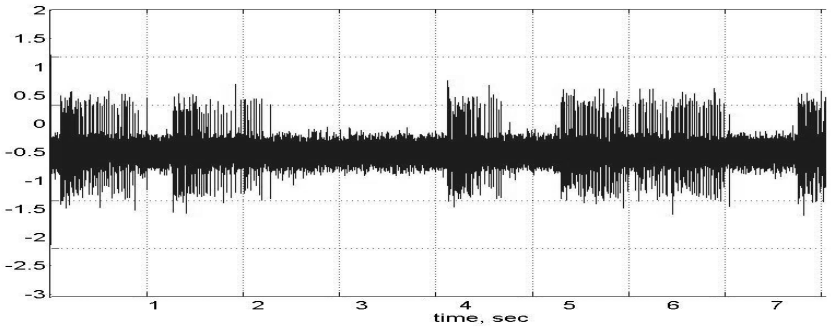
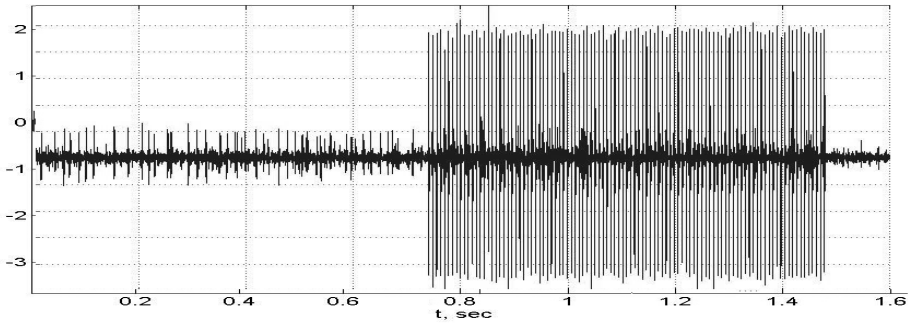


Figure 1. Record #2548, ASR = 2.2 : (a) signal of neuronal activity before and during DBS; (b) the same signal after filtering.

a)



b)

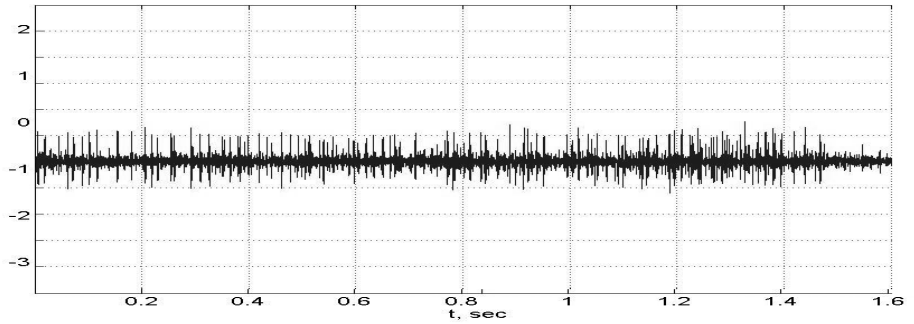
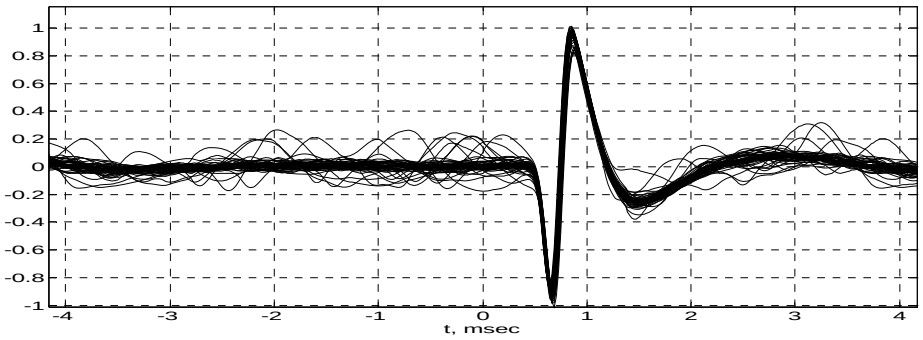


Figure 2. Record #S1529, ASR= 5.1: (a) signal of neuronal activity before, during and after DBS; (b) the same signal after filtering.

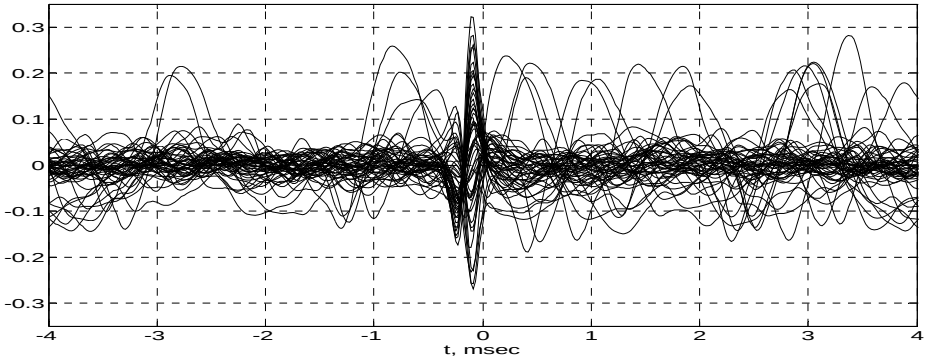
There are several recent studies focused on removal, subtraction, or filtering of stimulation artifacts from electrophysiological signals: electroencephalogram (EEG), electromyograms (EMG), local field potentials (LFPs) and extracellular microelectrode recording of neuronal action potentials.
Hardware techniques include hardware blanking (Robby&Lettich, 1975),(Knaflitz & Merletti, 1988) hardware filtering (Solomonow et al., 1985), varying amplifier gain (Roskar & Roskar, 1983) or their composition (Rossi et al., 2007). Algorithms of digital signal processing exploit several ideas. Time segments containing artifact peaks are cut off from the record at (O'Keeffee et al., 2001, Hines et al., 1996, etc.). An estimate the stimulus artefact waveform and subtracting from the signal is a basic idea of another group of algorithms.
An efficiency of the use of software or hardware blanking depends on the assumption that the stimulus artefacts and the biopotentials are well separated in time domain. In the case of extracellular microelectrode recording of neuronal action potentials during High-frequency DBS these techniques lead to loss of substantial part of information because of artifact duration is 20-50% of stimulation period while stimulus artifacts and action potentials are overlapping in the time. The overlapping of the stimulus artefacts and action potentials in frequency domain (Wichmann, 2000) leads to the loss of high frequency components of extracellular microelectrode recording of neuronal action potentials in case of the use of band pass filters.
To avoid the suppression of the high frequency components subtraction techniques were developed. However, most subtraction techniques suffer from an inability to adapt to the nonlinear dynamic of artefacts and hence suffer from the residual artefacts (O'Keeffee et al., 2001). Stimulus artifact suppression by subtraction of locally fitted polynomials is proposed in (Wagenaar & Potter, 2002). An advantage of this approach is that it doesn't depend on the stability of stimulus artefact waveform. Each artefact is considered as independent. This method enables to reduce a time interval after stimulation when action potentials can not be detected to 2 ms. Nevertheless high frequency interval of stimulus artefact (about 25% of record time for DBS 130Hz) is lost.
Estimates of representative stimulus artefact waveform was studied by Wichmann (2000). Template is generated by averaging a set of peri-stimulus segments adjusted by time shifting and scaling. Hashimoto et al. (2002) create several templates in order to account for artifact variability. Residual artefacts of 0.8-0.9 ms duration (Hashimoto et al (2002), and Tai et Al (2004)) are cut from the signal before further analysis. This part (about 11% of record time for DBS 130Hz) is no longer available for spike sorting, etc. Such a cut-off can still significantly corrupt spike trains obtained from the analyzed records.

This paper presents an algorithm for filtering the signal of neuronal activity during DBS. It is based on the use of a nonlinear oscillation model to explain variability of the artefacts of stimulation and provides signal synchronization in phase space

a)



b)



c)

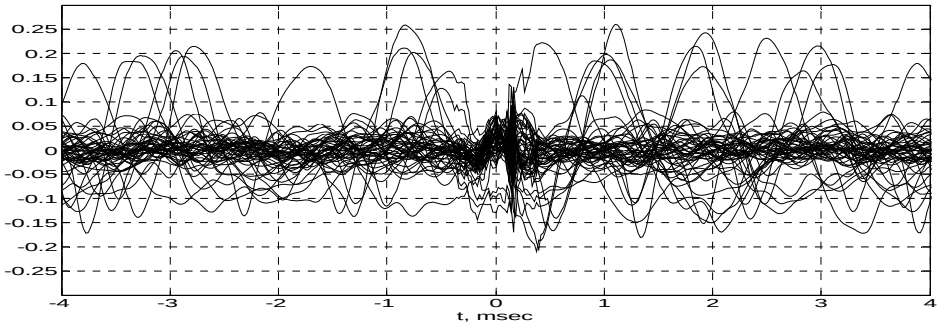


Figure 3. Signal #S1529, ASR= 5.1 sliced into the stimulation-period windows (a) before filtering; (b) after filtering based on the model with additive noise; (c) after filtering according to phase-space algorithm.

2. Methods

2.1. Signal Characteristics

The following indices of original and processed signal are used. *Artifact level* AL is a mean of maximal (original or processed) signal absolute values, such maxima were taken across artifact peaks. *Spike level* (SL) is a mean of maxima taken in spikes with no stimulation. Confidence interval (at confidence level 0.95) in assumption of Gaussian distribution of noise was used to estimate *Noise level*. The centered signal of neuronal activity observed before stimulation was used for its estimation. Because of presence of outliers in the data due to spikes, confidence intervals were estimated using median

absolute deviation MD. In case of normal distribution, the (theoretical) standard deviation σ and the absolute median deviation μ are linked via equation:

$$\Pr(|t| \le \mu) = \frac{2}{\sqrt{2\pi}} \int_0^{\mu} e^{-\frac{t^2}{2\sigma^2}} dt = \frac{1}{2}$$

Solving this equation we obtain $\mu = a\sigma$, where $a \approx 0.67448975$. Therefore, for a given confidence level we can use an estimation NL≅ (1.959965/*a*)*MD. From these values ASR=AL/SL, artifact-noise ratio ANR=AL/NL, spike-noise ratio SNR= SL/NL were derived.

2.2. Modelling

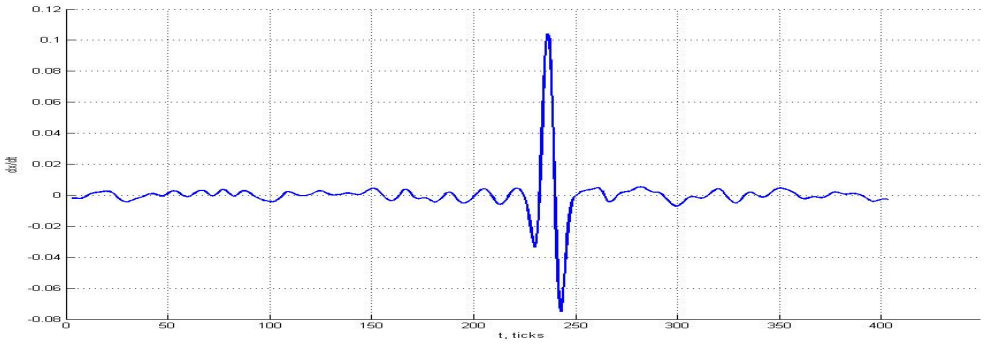
Basic methods (Hashimoto et al, 2002, 2003) treat artifacts as a periodic function with additive noise*:*

$$x_{St}(t + kT) = x_{St}^0(t) + \xi_k(t), \quad 0 < t \le T, \tag{1}$$

where *T* is the period of stimulation; *k* is the number of stimuli; $x_{St}^0(t)$ is the mean stimulus*;* ξ_k *(t)* are independent random variables with zero mean and finite variance $\sigma_\xi^2(t)$. In this case, the mean artifact can be estimated by averaging the observed signals of stimulation. Subtracting this mean from the signal can achieve relatively rough signal cleaning. The results of filtering according to Eq.1 are presented in Fig. 3b. The signals presented as examples in the articles (Hashimoto et al, 2002, Tai et al, 2004) have an ASR approximately equal to 3 - 6. After template subtraction, they still contained artifact residuals 1.5-2 times greater than spikes.

The quality of filtering is characterized also by the phase dependent standard deviations of the residuals *STD(t), 0 < t ≤ T* (Fig. 4).

a)



b)

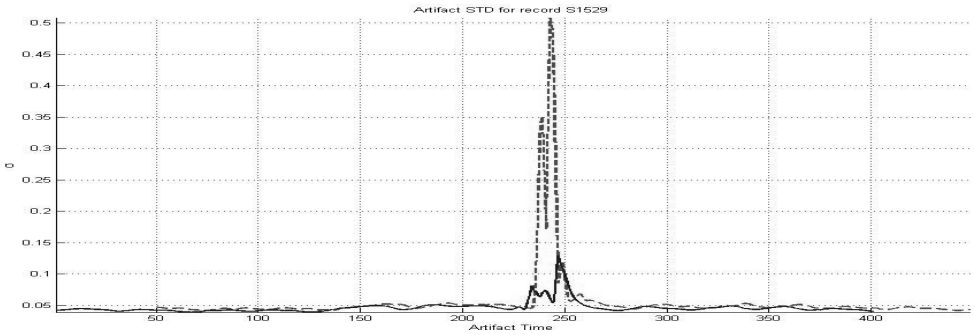


Figure 4. Record #S1529: a) derivative of mean artifact of stimulation; b) phase-dependent artifact STDs based on the representation of signal by the model with additive noise (Eq.1) (dash line) and by the nonlinear model of oscillations (continuous line) (Eq.2).

We note that the maxima of STDs correspond to the extrema of the derivative of artifact when the signal changes fast (Fig 4). Thus the reason for the loss of quality of filtering could be the loss of synchronization. In the present paper, we propose an approach for DBS signal filtration based on the synchronization in phase space.

We consider the observed signal $x(t)$, $t = 1,2,\ldots$ as a sum of the stimulation artifacts $x_{St}(t)$ and the signal $x_{Nr}(t)$ of neuronal activity: $x(t) = x_{St}(t) + x_{Nr}(t)$. Observations are available for discrete time moments. So, these signals can be considered to be smooth enough for the following model. The signal of stimulation is assumed to be a solution of an ordinary differential equation with perturbation

$$\frac{d^n x_{St}}{dt^n} = f\left(x,\ldots,\frac{d^{n-1}x_{St}}{dt^{n-1}}\right) + F\left(x_{St},\ldots,\frac{d^{n-1}x_{St}}{dt^{n-1}},t\right), \tag{2}$$

where n is the order of the equation, $F(\,)$ is a perturbation function, and equation

$$\frac{d^n x_{St}}{dt^n} = f\left(x_{St},\ldots,\frac{d^{n-1}x_{St}}{dt^{n-1}}\right) \tag{3}$$

describes a self-oscillating system with stable limit cycle $\mathbf{x}^0(t) = (x_1^0(t),\ldots,x_n^0(t))'$, $0 < t \le T$, in phase space with co-ordinates $x_1 = x_{St}, x_2 = \frac{dx_{St}}{dt},\ldots, x_n = \frac{d^{n-1}x_{St}}{dt^{n-1}}$. Here T is the period of stable oscillations which is the period of stimulation. The perturbation function $F()$, bounded by a small value, is a random process with a zero mean and a correlation time Δt_{corr} which is small in comparison to the period of stable oscillations: $B(F(\cdot, t), F(\cdot, t+\Delta t)) \approx 0$ if $\Delta t > \Delta t_{corr}$, $\Delta t_{corr} << T$.

In the case of stable oscillations (Eq. 3), the trajectory of the signal tends to the limit cycle whenever it is found in the neighbourhood. It provides the periodic solution if the initial point locates on the limit cycle. The perturbation function $F()$ in Eq. (2) tends to displace the trajectories of the signal out from the limit trajectory. However, if the perturbation is small enough, the trajectories stay in the neighbourhood of the limit cycle $\mathbf{x}^0(t)$, $0 < t \le T$ i.e., the solutions of Eq. (2) are similar to one another but they never coincide.

Let us introduce the local coordinates in the neighbourhood of the limit cycle following Gudzenko (1962). Let us fix an arbitrary point on the limit cycle P_0 as a starting point. The position of any arbitrary point P on the limit trajectory can be described by its phase θ, which is a time movement along the limit cycle from a starting point P_0, defined as $P_0(\theta = 0)$. Let us assume that function *f()* in Eq. (2,3) is twice continuously differentiable on all the arguments of the function, thus providing a necessary smoothness. At a point P with phase θ it is possible to construct a hyperplane (and only one such hyperplane) that is normal to the limit cycle. Let us consider an arbitrary trajectory Eq. (2) in the neighbourhood of the limit cycle. Denote $M(\theta)$ the point of its intersection to the hyperplane of phase θ and set the point of phase zero $M(0)$, $\theta = 0$ as the initial point for analyzed trajectory. Any trajectory can be described by variables $\mathbf{n}(\theta)$ and $t(\theta)$ (Fig. 5) where $\mathbf{n}(\theta)$ is a vector of normal deviation defined by $M(\theta)$ and its orthogonal projection $P(\theta)$ on the limit cycle. The second variable $t(\theta)$ is a time movement along the trajectory from the initial point $M(0)$ to the analyzed point $M(\theta)$. Thus, the limit trajectory is defined by $\mathbf{n}(\theta) \equiv \mathbf{0}$ and $t(\theta) \equiv \theta$, where $\mathbf{0}$ is a vector with all components equal to 0.

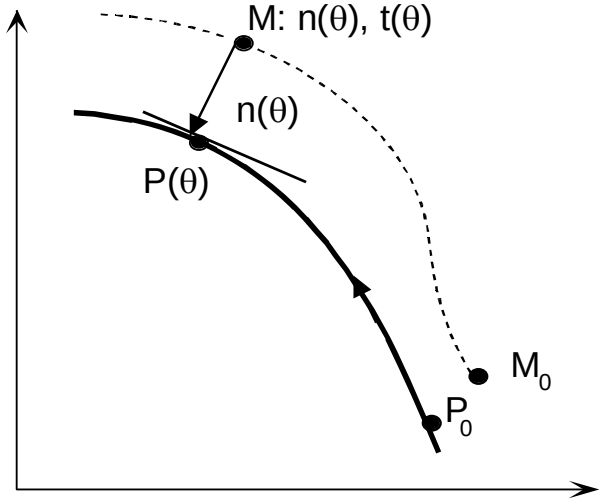


Figure 5. The local coordinates in the neighbourhood of stable limit cycle.

Let $\gamma(\theta)$ be the phase deviation, $\gamma(\theta) = t(\theta) - \theta$. Eq. (2) could be rewritten in linear approximation (Gudzenko, 1962) in the deviations $\mathbf{n}(\theta)$ and $\gamma(\theta)$ as follows:

$$\begin{aligned} \mathbf{n}(\theta)/d\theta + \mathbf{N}[\mathbf{n}] &= \mathbf{F}_n(\theta) \\ d\gamma/d\theta + (\Theta \cdot \mathbf{n}) &= \mathbf{F}_\gamma(\theta). \end{aligned} \qquad (4)$$

Here $\mathbf{N}(\theta)$ and $\Theta(\theta)$ are the functions of the parameters. As a result the signal trajectory in phase space is presented in linear approximation as a sum of periodic components of limit cycle and function of deviation $\mathbf{x}(t(\theta)) = \mathbf{x}^0(\theta) + \mathbf{n}(\theta)$, where $\mathbf{n}(\theta)$ and $t(\theta) = \gamma(\theta) - \theta$ are following Eq.4.

Thus, both models of the artifacts (Eq. 1 and Eq. 2) describe signals close to periodic ones. The model with additive noise (Eq. 1) explains the distortion of the amplitude of the signal only, while the model of nonlinear oscillations with perturbations (Eq. 2) explains the distortion of both amplitude and phase (Eq. 4).

For the following artifact filtering, the limit cycle that presents a periodic component of signal should be estimated. The limit cycle is described as $\mathbf{x}^0(\theta)$, $0 < \theta \le T$, phase running from *0* to *T*. Let us consider the segments of an arbitrary signal trajectory $\mathbf{x}^i(t(\theta))$, $0 < \theta \le$

T with phase θ running from 0 to T, referred to as cycles. General population $X = \{\mathbf{x}^i(t(\theta)), 0<\theta \le T\}$ is formed by cycles and corresponds to the realizations of the artifacts of stimulation. The limit trajectory $\mathbf{x}^0(\theta)$, $0 < \theta \le T$ corresponds to an ideal "undisturbed artifact of stimulation". The arbitrary cycle is presented as

$$\mathbf{x}^i(t(\theta))=\mathbf{x}^0(\theta) + \mathbf{n}^i(\theta)\ , 0<\theta \le T \tag{5}$$

where $\mathbf{n}^i(\theta)$ is determined by (Eq. 4). The mean trajectory converges to the limit cycle $\mathbf{x}^0(\theta)$ in linear approximation if the number of averaged trajectories increases infinitely (Gudzenko, 1962). It allows estimating the limit cycle $\mathbf{x}^0(\theta)$ by calculating the mean of the observed cycles $\mathbf{x}^i(t(\theta)), 0<\theta \le T$ in the phase space.

$$\tilde{\mathbf{x}}^0(\tilde{\theta}) = \frac{1}{k}\sum_{i=1}^{k}\mathbf{x}^i(t_i(\theta)), \quad \tilde{\theta} = \frac{1}{k}\sum_{i=1}^{k} t_i(\theta). \tag{6}$$

Other statistics could be also used, taking into account that $\mathbf{n}^i(\theta)$ and $\gamma^i(\theta)$ are characterized by an asymptotically Gaussian distribution for any θ in the case of uncorrelated noise $F()$ (Gudzenko 1962).

After the estimation of the mean of the artifacts of stimulation $\tilde{\mathbf{x}}^0(\tilde{\theta})$ in phase space, it is subtracted from the signal. For this purpose, local coordinates $t^i(\tilde{\theta})$ are calculated for each cycle. Then the functions of the vectors of normal deviations $\mathbf{n}^i(\tilde{\theta}) = \mathbf{x}^i(t^i(\tilde{\theta})) - \tilde{\mathbf{x}}^0(\tilde{\theta})$, $0<\tilde{\theta} \le T$ represent the signal after filtering in phase space. The first coordinates of the normal deviations represent the residuals of the artifacts in the time domain $x_{filt}(t) = n_1(t) + x_{Nr}(t)$, $t=1,2,\ldots$. Spline interpolation is used for regular partition in time.

2.3. Algorithm and its implementation

The implemented algorithm consists of following stages: (1) Unsupervised learning procedure; (2) On-line filtering.

Unsupervised learning procedure.

1. *Approximate the signal trajectories in phase space and detect artifacts.* For this purpose we compute the smoothed signal and the approximation of signal derivatives using Gaussian convolution. So, the derivatives $x^{(k)}(t)$, $k=0,1,\ldots, n-1$ are approximated by:

$$D^k x(t) = \int_{-\infty}^{\infty} x(s)\frac{d^k}{dt^k}\left(\exp\left(\left.(t-s)^2\right/2\alpha^2\right)\right)ds \tag{7}$$

   Artifact is detected by smoothed signal $D^0x(t)$ threshold crossing. The maximum over exceed threshold interval is chosen to be a center t^i of a segment containing a separate artifact $\mathbf{x}^i(t)$, $t \in [t^i-T/2, t^i+T/2]$.
2. *Collect N periods of stimulation to the training set* $X_N = \left\{\mathbf{x}^j(t)\right\}_{j=1}^{N}$
3. *Estimate a limit cycle.* In case of a symmetrical unimodal distribution density, the mathematical expectation $\mathbf{x}^0$ provides the maximum of $P(d(\mathbf{x}, \mathbf{x}^0)<R)$ for any given parameter R. Here $d(\mathbf{x}, \mathbf{x}^0)$ is a distance. Then, the element from training set $\tilde{\mathbf{x}}^0$ that provides the maximum of probability density in the neighbourhood were used to estimate the limit cycle (Aksenova et al., 2003)

$$\tilde{\mathbf{x}}^0 = \arg\max_{\mathbf{x}\in X_N}\left|\{\mathbf{y} \in X_N : d(\mathbf{x},\mathbf{y}) < R\}\right|, \tag{8}$$

where $|\cdot|$ is the number of elements of a set. Euclidian distance in $\Re^{nT}$ was employed as $d(\mathbf{x},\mathbf{y})$

$$d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\|, \qquad \|\mathbf{x}\| = \left(\sum_{\theta} \sum_{0 \le j \le n-1} \left(\frac{d^j x(t(\theta))}{dt^j} \right)^2 \right)^{1/2}, \quad \theta = 1,...,T. \tag{9}$$

The function of synchronisation $t^j(\theta)$, $0<\theta \le T$ of observed cycle $\mathbf{x}^j(t)$ to limit cycle $\mathbf{x}^0(\theta)$ is not known. Therefore the distance between two cycles $\mathbf{x}^i(t)$ and $\mathbf{x}^j(t)$, $0<t \le T$ was approximated by

$$d^2(\mathbf{x}^i,\mathbf{x}^j) = \sum_{t=0}^{T} \delta^2(t), \quad \delta(t) = \min_{\tau \in (-\tau_{\max},\tau_{\max})} \left\| \mathbf{x}^i(t) - \mathbf{x}^i(t+\tau) \right\|, \tag{10}$$

using a spline interpolation of the signals. Here τ_{max} is a maximum of phase deviation.

The time scale of the estimated limit cycle $\tilde{\mathbf{x}}^0$ was considered as a phase.

On-line filtering has the following steps.

1. *Approximate the signal trajectories in phase space and detect i-th artifact* $\mathbf{x}^i(t)$.
2. *Synchronize artifact* with the limit cycle by computing
$$t^i(\theta) = \underset{(\theta-\tau_{max},\theta+\tau_{max})}{\arg\min} \; d(\theta) = \underset{t \in (\theta-\tau_{max},\theta+\tau_{max})}{\arg\min} \left\| \mathbf{x}^i(t) - \tilde{\mathbf{x}}^0(\theta)) \right\|. \tag{11}$$
3. *Subtract the mean cycle* from each artifact in phase space, according to the synchronization.
4. *Present the result in time domain*.

On-line filtering is made for training set and for the following recoding.

The parameters of algorithms were chosen as follows.

Degree of the model. According to the model there are no points of self-intersections on the limit cycle in phase space. Only one normal hyperplane is constructed at each point of limit cycle for parameterization. Thus the degree of the model *n* was set to avoid points of self-intersections. For the records #S1529 and #2548 *n=3* allows observing (Fig. 8) the limit cycles without self-intersection in the region of peak of artifact

$$\frac{d^3 x_{St}}{dt^3} = f\left(x, \frac{dx_{St}}{dt}, \frac{d^2 x_{St}}{dt^2} \right) + F(\cdot).$$

Nevertheless because of presence of low frequency component in the signal to avoid additional filtration the model

$$\frac{d^4 x_{St}}{dt^4} = \tilde{f}\left(\frac{dx_{St}}{dt}, \frac{d^2 x_{St}}{dt^2}, \frac{d^3 x_{St}}{dt^3} \right) + F(\cdot)$$

was used. It provides less variance of trajectories in phase space because of Gaussian convolution applied for numerical differentiation act as a band pass filter.

Parameter of smoothness in Gaussian convolution. We decided to apply the same parameter α for all derivative approximations in (Eq.7). This strategy can assure that approximations of ODEs are identical to the original equations up to second-order terms. We use transfer functions as a criterion of suitable α. For operators D^k , the transfer functions have the following form:

$$\varphi^{(k)} = const \cdot \omega^{2k} e^{-\alpha^2 \omega^2} \quad (12)$$

Location of its maximum is $\omega_k^* = \frac{\sqrt{k}}{\alpha}$. We use a characteristic frequency calculated from the mean of the maxima $f_c = \frac{1}{3 \cdot 2\pi} \sum_{k=1}^{3} \omega_k^*$. This leads to an optimal $\alpha = 6.25 * 10^{-5}$ sec and f_c=3.52 kHz. This value of f_c agrees the best with the observed spectral band for spikes and artifacts. In the analyzed signals, most energy is concentrated in the band from 0.5 to 8 kHz. In order to simplify the calculations of convolutions, we restrict α to integer number of sampling ticks.

Size of training data set. In order to find a suitable training sample size *N* median behaviour with respect to *N* was explored. Mean distance to the sample median for different *N* were calculated for two reference records #2548 and #S1529. *N = 1000* provided stabilisation of mean distance and was selected as a training set size. Results of this experiment are depicted in the fig 6.

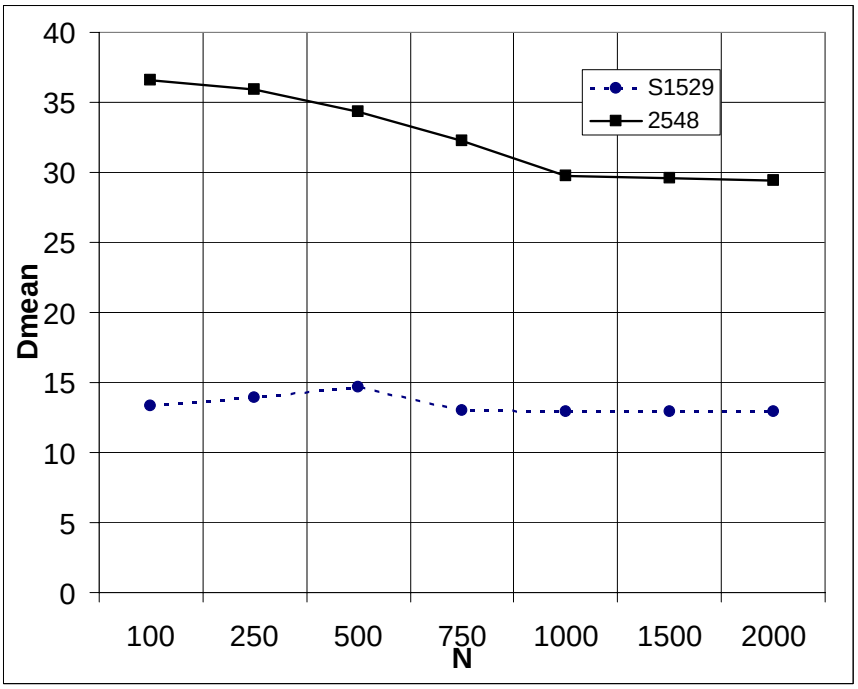


Figure 6. Mean distance to the sample median depending on training size *N* for two reference records #2548 and #S1529.

Threshold for artifact detection is set by user.

Maximal phase deviation. According to the model described above phase deviation $\gamma(\theta)$ has asymptotically normal distribution with variance that increases with time motion through limit cycle from starting point of parameterization θ_0 (Gudzenko 1962). In the algorithm synchronisation is carried out for each cycle of oscillation. Thus variance of phase deviation $\gamma(\theta)$ reaches their maximum for $\theta - \theta_0$ equals to period of oscillation. If time of the artifact peak is considered as a starting point two subsequent peak moments provide the maximal variance of phase deviation. Thus sampling distribution of times of

the artifact maximal peak were used to estimate maximal phase deviation τ_{max} . The difference between two subsequent peak moments equals

$$\Delta t_p(k) = T + \gamma^k(\theta_p) - \gamma^{k-1}(\theta_p) \quad (13)$$

where θ_p is the phase of peak and $\gamma^k(\theta_p)$ is the phase deviation for the *k-th* artifact. Taking into account the short correlation time of the perturbation function, we considered $\gamma^k(\theta_p)$ as independent random variables with a variance $\sigma_p^2 = \sigma^2(\theta_p)$. In this case it holds $\sigma^2(\Delta t_p) = 2\sigma_p^2$. We estimated $\sigma(\Delta t_p)$ directly from a set of peak times. Then the maximal phase deviation was fixed as $\tau_{\max} = 3\sigma_p$. In the case of asymptotically normally distributed $\gamma^k(\theta_p)$, it corresponds to the confidence level 0.997. For example, for the signal #S1529 we obtained σ_p=2.5927·10^{-5} sec. The maximal phase deviation was fixed as $\tau_{max} = 3\sigma_p =$ 7.7782·10^{-5} sec. Then $\tau_{\max}$ was adjusted by rounding to the nearest integer number of sampling ticks (4 ticks in case of sampling rate 48KHz).

Basic radius R. The radius parameter is calculated according to the variance of the observed cycles of artifact. Namely the sampling fractiles are used. As we have mentioned above, the cycles $\mathbf{x}(\theta)$, $0<\theta\le T$ could be considered as normally distributed in the Euclidian space $\Re^{nT}$. The difference of two independent **x** and **y** is also Gaussian. Thus the random values $\sqrt{2}(\mathbf{x} - E\mathbf{x})$ and **z** = **x**-**y** are identically distributed as well as the Euclidian distances $d^2(\mathbf{x}, E\mathbf{x}) = \|\mathbf{x} - E\mathbf{x}\|^2$ and $d^2(\mathbf{x},\mathbf{y})/2 = \|\mathbf{z}\|^2/2$. While the expectation of cycles is not known, we estimate the distribution density function of $d^2(\mathbf{x},\mathbf{y})/2$. For this purpose, we generated random pairs of cycles from the training set and we built a histogram. The position of a maximum was taken as a basic radius *R*, that corresponds approximately to the fractiles $q_{0.34} - q_{0.35}$. To calculate the fractiles, we approximate the histograms by χ_d^2 distributions. The effective degrees of freedom *d* were estimated as 7-8, which leads to the fractiles mentioned above. So 34.1 – 35.4% of cycles have to fall in the *R*-ball of limit cycle. In our experiments, we obtained 35-40% of cycles in the ball of radius *R* by direct count.

3. Experimental Results

3.1 Data Description

Real Data Description. Eight records collected from five human patients during DBS surgery were used to test the algorithm. Neuronal activity is recorded before, during, and after high-frequency stimulation within the *subthalamic nucleus*, *Globus Pallidus* or *Substantia Nigra* (for some records the post-stimulation segment is not available). Signals were obtained using Medtronic external stimulator, and captured using AlphaOmega recording system. They were measured in volts after amplification. The records with a sampling rate of 48 kHz have 9 to 100 seconds duration, stimulation lasting 7-80 seconds. Periodic stimuli were delivered through microelectrodes placed 2mm apart in the same brain nucleus, with a frequency at 130Hz (i.e. with period approx 370 sampling ticks) and pulse duration 60 μs. Pulse intensities were in the range of 1000 μA - 6000 μA.

Synthetic Dataset. In order to explore algorithm's behaviour depending on artifact amplitude we constructed a set of artificial signals with predefined ASR. To the record (6-second segment) containing artifacts with no neuronal activity another signal with

single-neuron spikes and no stimulation were added. Appropriate constant K gives desired value of ASR:

$$X_{Syntethic}(t) = X_{Artifacts}(t) + K\, X_{Single\text{-}Neuron}(t).$$

Collection of such signals for ASR from 2 to 20 was generated.

3.2 Tests of the Algorithm

To compare the quality of signal description by the standard model with additive noise (Eq.1) and model (Eq.2) the appropriate phase dependent standard deviations (Fig. 4) of the residuals $STD(t)$, $0 < t \le T$ were calculated. We estimated standard deviations $\sigma_{\xi}(t)$, $0 < t \le T$ of additive noise $\xi_k(t)$ in the model (Eq.1) and $\sigma_{n_1}(\theta)$, $0 < \theta \le T$, standard deviations of $n_1(t)$, the first coordinates of the vectors of normal deviation (Eq.5). Values $n_1(t)$ represent the residuals of artifacts in the time domain for the model (Eq.2).

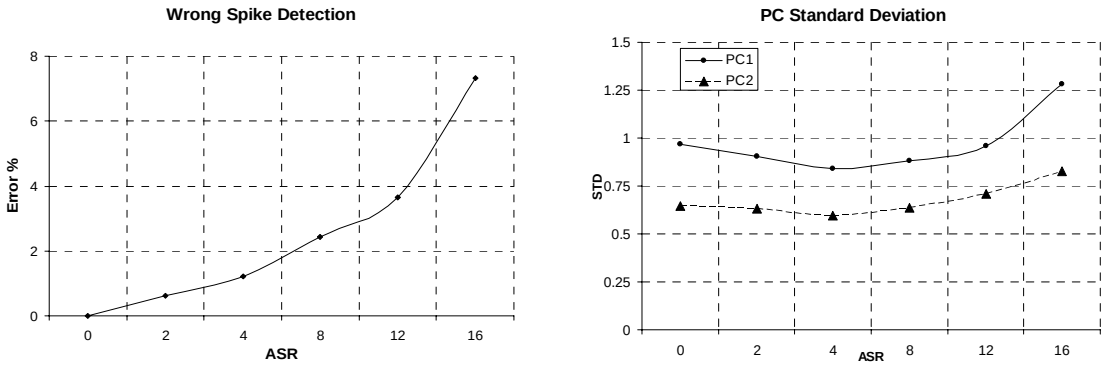
Eight real data records described above were used to estimate $STD(t)$. The phase model (Eq.2) produces much smaller variances especially close to the peak (Table 1). The sums of STD across residual were in average 2.13 times less and the maxima of STD were 1.5-2.1 (1.8 in average) times less for the phase model.

Table 1: STD of artifact residuals in the artifact peak zone

Record #	***Filtering in time domain***		***Filtering in phase space***	
	Max. STD	*Sum of STDa across residual*	*Max. STD*	*Sum of STDa across residual*
2548	0.106611	4.34	0.057173	1.76
S1529	0.118543	2.57	0.058254	1.251
F096K	0.088063	6.22	0.018505	2.04
N0047	0.151444	3.52	0.095315	1.84
N0023	0.14377	3.96	0.072804	2.20
DUG1581	0.086437	3.86	0.051635	1.935
DUG4821	0.201742	3.88	0.147963	2.18
S467	0.037689	1.646	0.023234	0.822

Tests of the Algorithm with Synthetic data. The test intends to reveal what happens to spike trains and spikes itself in presence of stimuli and cleaning procedures. Spike trains obtained from an original signal of neuronal activity $X_{Single\text{-}Neuron}(t)$ before mixing with artifacts and after full processing were compared. Simple threshold filter was used for spike trains extraction. Percentages of spikes found in processed signal at their correct positions were measured. Let us note that summation of signals with coefficient K changes the spike-noise ratio. The graphs of errors of spike detection depending on ASR and corresponding SNR are depicted in the Fig. 7a, 7c. Fig. 7b represents variances of two first principal components of spikes depending on ASR. We can see that spikes rest practically intact up to ASR about 10-12.

a) b)



c)

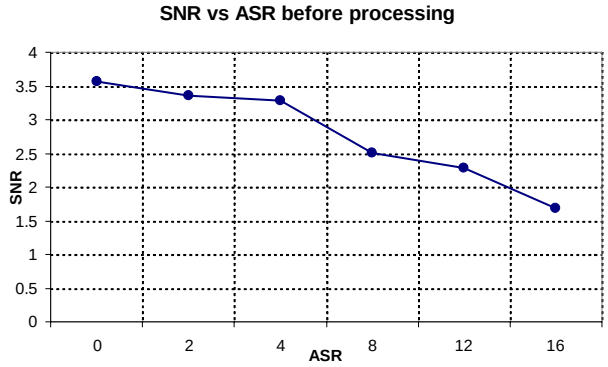
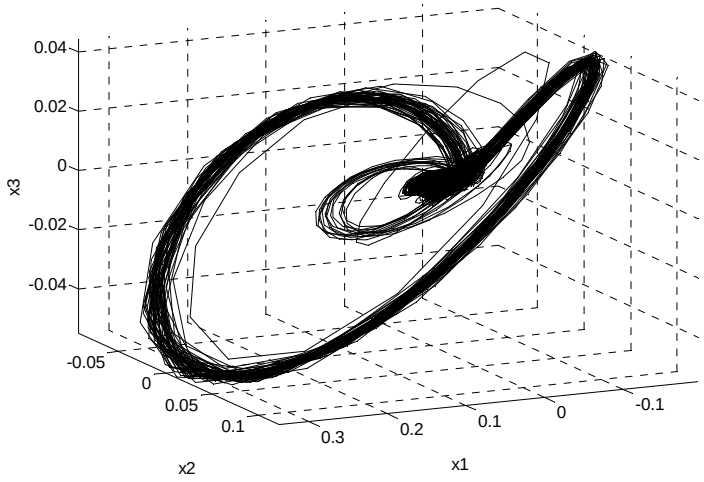


Figure 7. Post-filtering spike quality for synthetic data: a) percentage of errors of spike detection after filtering; b) variance of first two principal components for detected spikes; (c) SNR in synthetic signals used for the test. ASR=0 corresponds to original neuron activity with no stimulation added.

3.3 Real Data filtering

The records described above were used to test the algorithm on real data. The signals are characterized by ASR $\in$ *[1.3, 14]*, ANR $\in$ *[3, 160]* and SNR $\in$ *[1.4, 5.1]*. Examples of original and filtered signals are shown in Fig. 1,2. Fig. 3 display signal sliced into sets of stimulation-period windows, before and after filtering. The figure demonstrates the shape of the original artifacts and of the residuals left after full processing. One can see also the neuronal spikes. Phase portraits in 3D of appropriate artifact sets are depicted in the Fig. 8. In the Fig 8a we can see spike trajectories (smaller orbit) of record #2548.

a)



b)

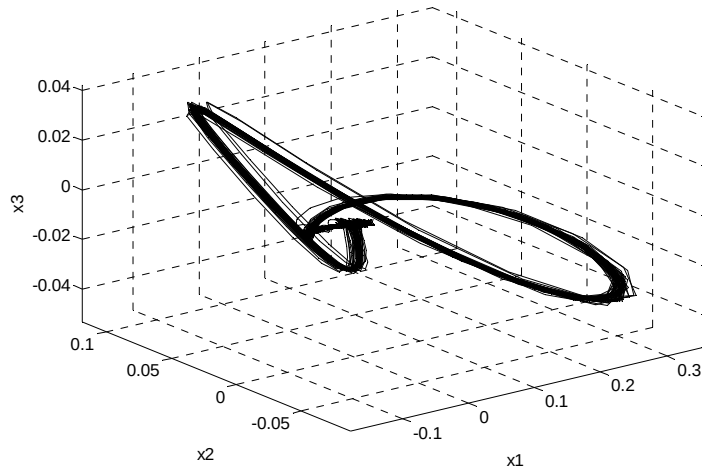


Figure 8. Artifact trajectories in the phase space projection with coordinates $D^1x(t)$, $D^2x(t)$, $D^3x(t)$ for records #2548 (a) and #S1529 (b).

Dependences between Artifact Spike Ratio before and after signals filtering, both in phase space and time domain are presented in the Fig 9. For comparison, we have also shown such results for synthetic data. As a result we can see curves being in good fit with real-data cases.

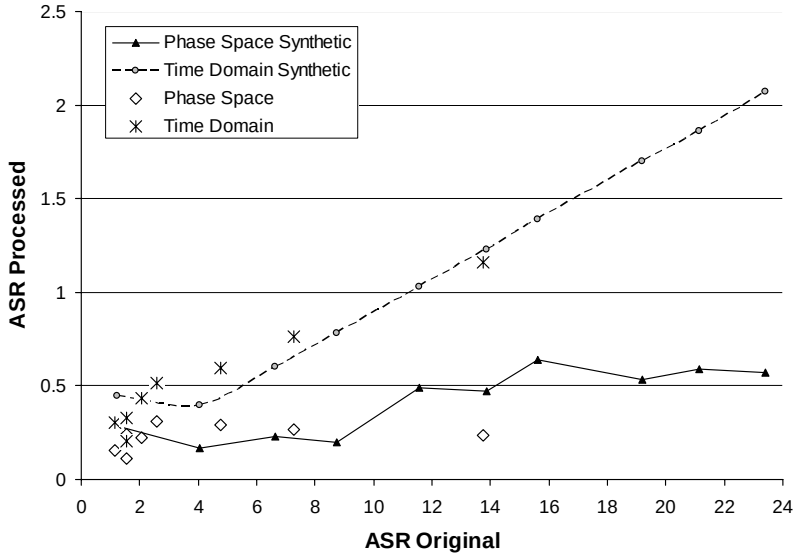


Figure 9. Residual-spike ratio depending on pre-filtering ASR comparing time-domain and phase-space algorithms for synthetic and real data. Results for eight real records (markers) and synthetic signals (dash and continuous lines) are in a good fit.

4. Discussion

The most commonly known algorithm for artifact subtraction (Hashimoto et al, 2002) is based on making artifact templates by averaging the set of peri-stimulus segments in time domain. After template subtraction 11% (DBS 130Hz) of record time are removed from the signal. This paper presents a new method for the filtering of the signal of neuronal activity during DBS with artifact template subtraction. It is based on nonlinear oscillation model with perturbation as a basic model of stimulus artifacts instead of model with additive noise. This model explains signal distortion in both amplitude and phase. The set of synchronised in phase space artifact is used to estimate 3D artifact template with mean or median. Template is subtracted from the signal according to synchronization. The use of the proposed method and algorithms allows a 2-3 times reduction of the residual artifact of stimulation in comparison with the standard model with additive noise. The tests with artificial signals that are the combination of real signals of neuronal activity and stimulus artifacts showed that spike trains are less corrupted. This promising approach will enable analysis of neuronal activity during DBS to more effectively study the mechanism and to therefore improve the DBS technique.

Let us note that all data in this paper were obtained using standard elecrophysiological equipment. Such devices limit suitable artifact amplitude because of their insufficient dynamic range. In future research, the use of dedicated (more precise) recording systems will allow studding the problem of filtering of stimulus artifacts of higher amplitude and variance.

Acknowledgment

This study was partially supported by a Medtronic Grant to INSERM U318. The authors wish to express their appreciation to Dr. Brigitte Piallat for providing the recording used in the studies.

References

1. Benabid A.L., Wallace B., Mitrofanis J., Xia R., Piallat B., Chabardes S., Berger F., A putative generalized model of the effects and mechanism of action of High Frequency Electrical Stimulation of the Central Nervous System, *Acta neurol. Belg*., 2005, 105 :149-157.

2. McIntyre C.C., Grill W.M., Sherman D.L., Thakor N.V., Cellular effects of deep brain stimulation: model-based analysis of activation and inhibition, *J.Neurophysiol*. 2004, 91:1457-1469.

3. Roby RJ, Lettich E. A simplified circuit for stimulus artefact suppression. Electroenceph. Clin. Neurophys. 1975;39:85–7.

4. Knaflitz M, Merletti R. Suppression of stimulus artifacts from myoelectric-evoked potential recordings. *IEEE Trans Biomed Eng*, 1988; **35**(9):758–63.

5. Solomonow M, Baratta R, Miwa T, Shoji H, D'Ambrosia R. A technique for recording the EMG of electrically stimulated skeletal muscle. *J. Orthoped*., 1985;8(4):492–5.

6. Roskar E, Roskar A. Microcomputer based electromyographic recording system with stimulus artefact suppression. *In: Third Medical Conference on Biomedical Engineering*, 1983. Yugoslavia, Portoroz.

7. Rossi L**,** Foffani G**,** Marceglia S**,** Bracchi F**,** Barbieri S**,** Priori A**.** An electronic device for artefact suppression in human local field potential recordings during deep brain stimulation. *J Neural Eng.* 2007 Jun;4(2):96-106. Epub 2007 Mar 13.

8. Blogg T, Reid WD. A digital technique for stimulus artifact reduction. Electroencephalogr. *Clin. Neurophysiol.* 1990;76:557-61

9. Winchman T. A digital averaging method for the removal of stimulus artifacts in neurophysiologic experiments. *J. Neurosci. Meth*, 2000 ;98:57–62.

10. Wagenaar D.A.,. Potter - S.M. Real-time multi-channel stimulus artifact suppression by local curve fitting - DA J. *Neurosci. Methods*, 2002

11. Derek T. O'Keeffe, Gerard M. Lyons, Alan E. Donnelly, Ciaran A. Byrne. Stimulus artifact removal using a software-based two-stage peak detection algorithm. *Journal of Neuroscience Methods*; 109, 2001, 137–145

12. Hines AE, Crago PE, Chapman GJ, Billian C. Stimulus artifact removal in EMG from muscles adjacent to stimulated muscles. *J. Neurosci. Meth.*, 1996; 64:55–62.

13. Hashimoto T, Elder C.M., Okun M.S., Patrick S.K., Vitek J.L., Stimulation of the subthalamic nucleus changes the firing pattern of pallidal neurons, *J.Neurosci*., 2003, 23 :1916-1923.

14. Hashimoto T, Elder CM, Vitek JL. A template subtraction method for stimulus artifact removal in high-frequency deep brain stimulation. *J Neurosci Methods*, 2002, 113:181-186.

15. Tai CH, Boraud T, Bezard E, Bioulac B, Gross C, Benazzouz A. Electrophysiological and metabolic evidence that high-frequency stimulation of the subthalamic nucleus bridles neuronal activity in the subthalamic nucleus and the substantia nigra reticulata. *FASEB J.* 2003 Oct;17(13):1820-30

16. Gudzenko, L.I., Statistical method for self-oscillating system characteristics detection, *Izvestiia Vuzov Radiophysics*, 1962, 5: 573-587.

17. Aksenova, T.I.; Chibirova O.K., Dryga, O.A., Tetko, I.V.; Benabid A.-L.; Villa,A.E.P.; An unsupervised automatic method for sorting neuronal spike waveforms in awake and freely moving animals. *Methods*, 2003, 30: 178-187.

Chapter 5

ARTICLE IN PRESS

Available online at www.sciencedirect.com
SCIENCE DIRECT®

Sensors and Actuators B xxx (2004) xxx–xxx

SENSORS AND ACTUATORS B CHEMICAL

www.elsevier.com/locate/snb

Preprocessing of matrix QCM sensors data for the classification by means of neural network

A.M. Reznik*, A.A. Galinskaya, O.K. Dekhtyarenko, D.W. Nowicki

The Institute of the Mathematical Machines and Systems, National Academy of Sciences of Ukraine, 42 Glushkov Ave, Kiev 03187, Ukraine

Abstract

An experimental comparison of linear and non-linear pre-processing methods for olfactory data is made. The original data are formed by 280 values of reaction from six quartz crystal microbalance (QCM) sensors taken at 1 s intervals. Data vectors are processed with the non-linear maximum filter or by the linear averaging filter and are used as inputs of a feedforward neural network using the back-propagation learning rule, one hidden layer containing from 5 to 15 neurons. The filter window size is 5–10.

The learning set is composed of 60 sensor reactions for six types of cologne. The neural network correctly classifies 82–86% of independent examples. The usage of the maximum filter with a small window size allows an increase of the classification rate by 3–5%. The best results (86%) are obtained when only first 50 measurements of sensor reaction are used.

Keywords: Neural networks; Odour recognition; QCM sensors; Data preprocessing

1. Introduction

The usage of neural networks for the odour recognition is based on their ability to identify the differences in response to the sensors reacting in different ways for various aroma substances [1,2]. The values of matrix sensor reactions analyzed by the neural network contain redundant components and fluctuations that complicate the learning process and distort classification results. In order to depress these factors data preprocessing is used which includes filtration and normalization to the $[-1, +1]$ range, corresponding to the neuron activation values. In a number of cases the effectiveness of a neural network could be increased with the help of the non-linear transform of input data. This phenomenon is caused by the change of Vapnik–Chervonenkis dimension and is used in SVM classifier [3], which resolves the most difficult classification problems where the data are not linearly separable. The same principle of non-linear preprocessing operations can be used to improve the classification of linearly inseparable data by a feedforward neural network.

At the present time there are two dominant methods of approach to the preprocessing of olfactory data classified by a neural network.

1. The training of a neural network with redundant architecture followed by the removal of elements of little information (connections and/or neurons).
2. The usage of the principle components technique to reduce the data redundancy at the input of a network.

In the first case, the preprocessing functions are laid on the neurons of the first hidden layer and their required functions are formed during the training process. The training of a neural net consists of a few phases. At the end of each phase the information density, i.e. the influence of each component (connection/neuron) on a behaviour of the entire network is estimated and the least significant components are removed. This method, also known as "pruning" [4], is effective in cases when the network learning process has a sufficient degree of convergence. Unfortunately, it is not always easy to find such a network.

The method of principal components (often referred as principal components analysis (PCA)) is based on an extraction of senior components of data decomposition into the basis formed by the correlation matrix eigenvectors. These senior components hold the most signal energy and the ratio of their amplitudes is often sufficient to identify the sensor reaction [5]. Usually, the method is used to increase information density and to decrease the dimension of input data fed to a neural network [6].

The PCA method exploits the model of Gaussian distribution of data. The experimental comparison [7] of the op-

* Corresponding author. Tel.: +340 44 2665548; fax: +340 44 2666457.
E-mail address: neuro@immsp.kiev.ua (A.M. Reznik).

0925-4005/$ – see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.snb.2004.05.047

timal classifier based on such a model along with the neural network trained with the same data shows that the neural network provides better classification results. It also finds more effective data representation than correlative dependencies used by the Gaussian model. One can conclude that the limitation of the number of principal components increases the risk of loosing some specific input data features whereas these features can be found by the neural network on its own.

Among new research to the given problem the work [8] should be mentioned that offers a method combining the two approaches named above. The PCA method is used to form the weights of the first hidden layer omitting the direct neural network training. Next, the network is trained and then the pruning procedure is used to delete the connections and the input components of little information. A combination of different approaches will help to decrease the probability of the training process being stuck in local minimum, and to decrease the degree of network specialization. Authors of [8] managed to decrease in hundreds of times the dimensions of input data and the number of sensors used. It is still not clear whether or not this approach remains effective for the classification of linearly inseparable data that requires large networks with dozens of neurons and/or a few hidden layers.

The aim of the given research is to discover the possibility of improvement of a neural network used to classify quartz crystal microbalance (QCM) odour data with the help of non-linear input data transform by the maximum filter. A feedforward neural network with one hidden layer is used with the QCM sensors [6] data fed into its input. The experiments were made using the NeuroLand [9] software with the experimental dataset used in [6] for the odour classification by means of associative memory.

2. The research formulation

In the problem, we consider the signals entering the network input are formed by the sequences of resonance frequency values measured from the QCM sensors. These values are decreased over time reflecting the pace of odour molecules absorption process. A neural network should disclose the differences in dynamics of these sequences in order to make the proper classification. The natural informational features revealed by the network are unknown, but one can conclude the introduction of non-linear input data transform could improve the efficiency of network usage. So as to verify this assumption experimentally, we used the maximum filter, which is a non-linear data transform via selecting the maximum value within a given observation window. In favour of this approach two points of argument can be pointed out.

Firstly, the effectiveness of a suggested approach can easily be verified by its comparison with the similar data obtained with the help of averaging within the same observation window.

Secondly, the probability distribution of transformed data that corresponds to the high-order rank statistics within the window of width H has a simple relationship with the probability distribution $P(x)$ of the original data:

$$F_H(x) = P(X_H \le x) = P^H(x) \tag{1}$$

This indicates that transformed data retain the statistical moments of high order, which, presumably, play an important role in formation of classification features used for the neural net training.

The basic obstacle for the direct usage of maximum selection method is the non-stationarity of QCM sensor responses with exponential time dependence. The ratio of subsequent reaction values was used in order to eliminate this dependence. It provides the equalization of average signal values at the input of neural network:

$$\overline{u(t)} = \frac{x(t)}{\overline{x(t-1)}} = \frac{\overline{x(t_0)}\exp\alpha(t_0 - t)}{\overline{x(t_0)}\exp\alpha(t_0 - t + 1)} = \exp\alpha, \tag{2}$$

where $\overline{x(t)}$ is the average value of sensor reaction at the moment t and α is the speed of sensor reaction change.

During preprocessing the sequence of measurements of each sensor was divided into equal intervals (observation windows) containing H measurements and for each of them the maximum values were found:

$$w(k) = \max[u(h)],\ kH \le h < (k+1)H \tag{3}$$

These maximum values were fed to the neural net input.Another set of experiments was carried out using the averaging filter instead of the maximum filter:

$$v(k) = \frac{1}{H}\sum_{h=kH+1}^{(k+1)H} u(h) \tag{3}$$

The task of research was to compare the results of network training using the maximum filter and the averaging filter. Experiments were carried out with different observation window sizes and different numbers of neurons in network's hidden layer. At the same time it was intended to estimate the degree of information density of various sensors' measurements and, particularly, to elaborate the results of experiments that were done [6].

3. Experiment description

3.1. Initial data

The data of quartz sensors based on the microbalance principle (QCM) were used during the experiment. These sensors are the quartz resonators with fundamental frequencies of about 10 MHz whose surface is covered by the sensible layers of various substances (calixarenes, etc). The experimental device had seven sensors of this type and an electronic unit that provides measurement and recording of

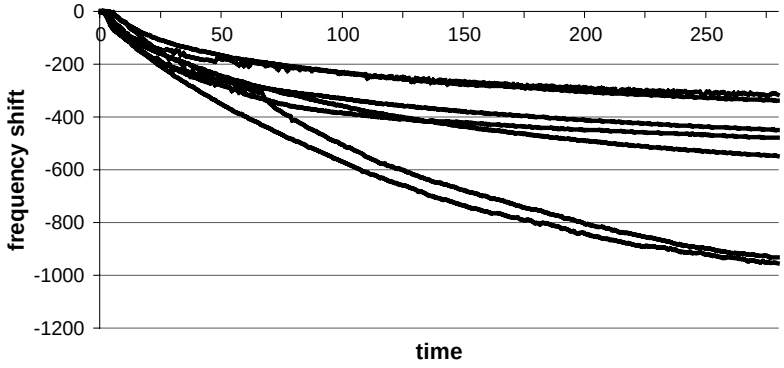


Fig. 1. Sensors' response for the ethanol sample.

sensors' fundamental frequencies with 1 s interval for the duration of 280 s. The result of each experiment is the sequence of 1960 integer measurement values. Fig. 1 depicts an example of such a sequence obtained for the ethanol sample.

The list of substances used and their total amount is shown in Table 1.

3.2. Neural network

Neural network of feedforward type contained one hidden layer with 5, 10 or 15 neurons. Data coming to each of the net inputs were normalized to the [0, 1] range. Normalizing coefficients were calculated using the training dataset. The neural network was trained using the standard online back-propagation algorithm. The learning rate was set to 0.05 and the moment coefficient was 0.15. Save Best Method was used during the training that lasted for 5000 epochs. Output neurons were used in classification mode, i.e. the network reaction was determined according to the maximum value of the post-synaptic potential. Experiments were carried out using the NeuroLand software package [9].

Taking into account a comparably small amount of input data, we used six variants of decomposition of initial dataset into train and independent test parts. These datasets were formed by random selection of 20 train examples from the initial data. Every variant of selection contained examples of all analyzed classes. Thus, for each reported classification result network training and testing was made six times with independent parts of data and final result was derived as the average of these tests.

Table 1
Classes of chemical images

Class	Substance	Number of images
1	Ethanol	11
2	Landish	5
3	Russki Les	8
4	Roksolana	6
5	Shipr	12
6	Siren	15
7	Water	3

4. Experimental results

The results of testing of the neural network with different number of hidden neurons are depicted in Figs. 2–4. The average percentage values of correctly classified odour patterns are shown for the maximum filter and the averaging filter depending on the observation window size. These results were obtained using three subsets of sensor reaction values: values collected during the first 50 s, collected in the interval 50–200 s, and all 280 values of reaction of each sensor.

Comparing the results obtained, one may note that as the sizes of hidden layer and observation window increases, the difference between two types of preprocessing vanishes. The greatest difference is observed with the window size of 3. The network with five hidden neurons plus using the first 50 values of sensor reaction had the highest results. The maximum classification value, about 86% of correct responses, was obtained using the maximum filter with the window size of 3. Under the same conditions the results of averaging filter were 6% lower. Networks with greater number of hidden neurons also exhibited better results with the maximum filter (about 83.5%).

The results of experiments estimating the influence of used normalization methods on the classification results are given on Figs. 5 and 6. These figures depict the dependencies of correct classification value on the number of hidden neurons for the original data and the data normalized using (1). The experiments were made with the complete data vectors containing 280 values and with truncated ones that contain reaction values collected for the first 50 s. The results for the maximum filter are shown on Fig. 5 and the results for the averaging filter are shown on Fig. 6. In both experiments the observation window size was set to 3.

For the normalized input data the best results—about 86% are obtained using the maximum filter with the network having five hidden neurons and truncated data vectors. Without normalization both the maximum and averaging filters

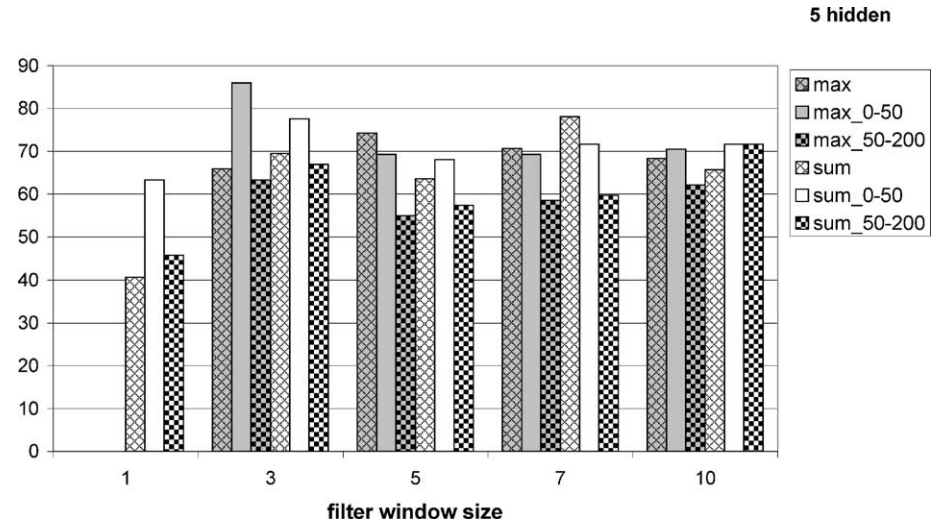


Fig. 2. Classification results for the network with five hidden neurons and normalized input data.

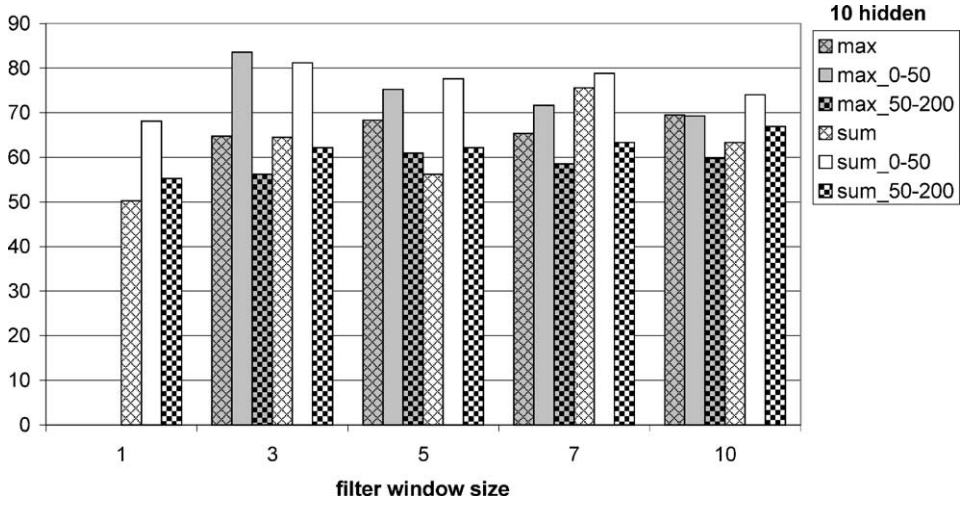


Fig. 3. Classification results for the network with 10 hidden neurons and normalized input data.

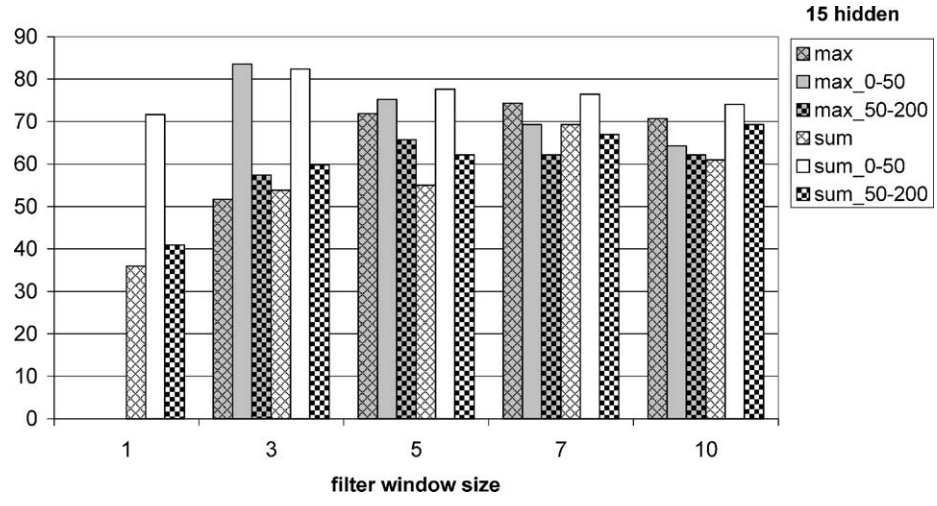


Fig. 4. Classification results for the network with 15 hidden neurons and normalized input data.

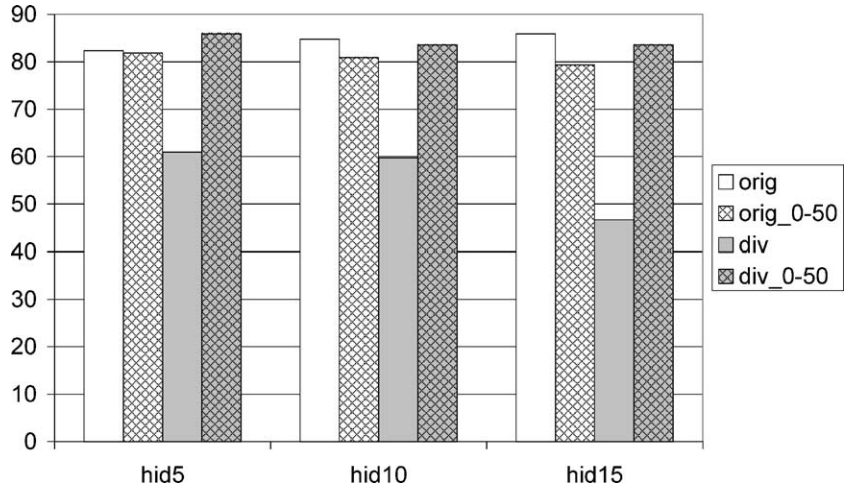


Fig. 5. Classification results for the maximum filter with window size 3.

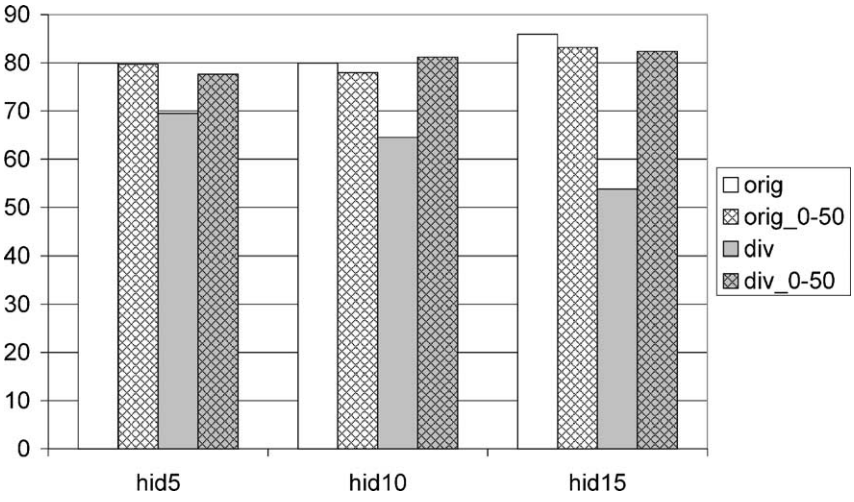


Fig. 6. Classification results for the averaging filter with window size 3.

achieved the same result, but it requires a network with 15 hidden neurons and complete data vectors.

The usage of normalization does not affect the results of the averaging filter but considerably worsens the maximum filter results in the case of using complete data vectors.

5. Discussion

Analyzing the results of experiments made it is necessary to take into account that the non-linear transform does not always improve the neural network operation. This transformation changes the character of the error surface thus influencing the speed and the results of neural network training. Its consequences could be positive as well as negative depending upon the character of data, the neural network architecture and its parameters.

The experimental results proved the assumption about the improvement of neural network training by the maximum filter and non-linear normalization of QCM sensor reactions. This improvement amounts up to 5% and is obtained for the network with five hidden neurons and rather small observation window (3 s) using the first part of sensor reactions (first 50 s). This is a quite good result taking into account that the same classification value can be obtained using the network three times as big (15 hidden neurons) that operates with the whole range of sensor reaction taken from 280 s intervals.

Acknowledgements

This work was supported by the INTAS grant no. 01-257.

References

[1] P.E. Keller, Overview of electronic nose algorithms, in: Proceedings of the International Joint Conference on Neural Networks, 1999.

[2] J.W. Gardner, P.N. Bartlett. Electronic Noses, Principles and Applications, Oxford University Press, Oxford, 1999.
[3] V. Vapnik, Statistical Learning Theory, Wiley, 1998, pp. 736.
[4] S. Haykin, Neural Networks: A Comprehensive Foundation, second ed., Prentice Hall Inc., 1999, pp. 842.
[5] I.V. Kruglenko, B.A. Snopok, Y.M. Shirshov, E.F. Venger, Digital aroma technology for chemical sensing: temporal chemical images of complex mixtures, Semicond. Phys., Quant. Electron. Optoelectron. 3 (4) (2000) 529–541.
[6] A.M. Reznik, Yu. M. Shirshov, B.A. Snopok, D.W. Nowicki, A.K. Dekhtyarenko, I.V. Kruglenko, Associative memories for chemical sensing, in: Proceedings of the International Conference on Neural Information Processing, Singapore, 2002.
[7] A.M. Reznik, A.A. Galinskaya, K.M. Kuzhel, On the information density of Gaussian components used for the classification of ultrasonic location signals, in: Proceedings of the Eighth All-Russian Conference on Neurocomputers and Their Applications, 21–22 March 2002, Moscow, Russia, pp. 244–250.
[8] Z. Boger, R. Cavicchi, S. Semancik, Analysis of conductometric microsensor responses in 36-sensor array by artificial neural networks modeling, in: Proceedings of the International Symposium on ISOEN'02, Rome, 2002.
[9] A.M. Reznik, E.A. Kalina, A.S. Sitchov, E.G. Sadovaya, O.K. Dekhtyarenko, A.A. Galinskaya, multifunctional neurocomputer NeuroLand, in: Proceedings of the International Conference on Inductive Simulation, vol. 1 (4), Lviv, Ukraine, 2002, pp. 82–88.

Biographies

A.M. Reznik, Head of Department of Neural Technologies in Institute of Mathematical Machines and Systems, received degree in technical sciences in 1998, PhD in technical sciences in 1968 and MS in Kiev Polytechnic Institute, Electro-Acoustical Department in 1958. The fields of interest are object recognition, acoustics and neural networks.

A.A. Galinskaya, Researcher, Institute of Mathematical Machines and Systems. She is a post-graduate student in Institute of Mathematical Machines and Systems, NAS Ukraine since 2002, received MS in informatics, National Technical University of Ukraine "Kiev Politechnical Institute" in 2002 and BS in applied mathematics, National Technical University of Ukraine "Kiev Politechnical Institute" in 2000. The fields of interest are artificial neural networks, system programming, mathematical modeling, data processing and pattern recognition.

O.K. Dekhtyarenko, Researcher, Institute of Mathematical Machines and Systems. He is a post-graduate student at Educational and Scientific Center of Physics and Technology, National Academy of Science of Ukraine. He received his MS in applied informatics at National Technical University of Ukraine "Kiev Polytechnic Institute", Department of Physics and Technology in 2002 and a Bachelor's degree in applied mathematics in 2000. The fields of interest are neural networks, data processing, object-oriented analysis, programming and design.

D.W. Nowicki, Researcher, Institute of Mathematical Machines and Systems. He is a PhD student at Universite Paul Sabatier Toulouse, France. PhD degree in applied mathematics expected in 2005. Also, is a PhD student at Moscow Institute of Physics and Technology (Kiev Subdivision), PhD degree in theoretical computer science expected in 2003. MS in applied mathematics was received at Moscow Institute of Physics and Technology in 2000 BS in applied mathematics and physics, Moscow Institute of Physics and Technology in 1999. The fields of interest are neural networks, artificial intelligence, mathematical programming, data mining, time series analysis, natural language processing, non-linear dynamics, graph theory, theoretical computer science, routing algorithms, stochastic processes, simulation techniques, etc.

ASSOCIATIVE MEMORIES FOR CHEMICAL SENSING

Reznik A.M.[1], Shirshov Yu. M.[2], Snopok B.A.[2],
Nowicki D.W.[1], Dekhtyarenko A.K.[1], Kruglenko I. V.[2]

[1]Institute of Mathematical Machines and Systems of NAS of Ukraine
[2]Institute of Semiconductor Physics of NAS of Ukraine
neuro@immsp.kiev.ua

ABSTRACT

We consider application of neural associative memories to chemical image recognition. Chemical image recognition is identification of substance using chemical sensors' data. The primary advantage of associative memories as compared with feed-forward neural networks is high-speed learning. We have made experiments on odour recognition using hetero-associative and modular auto-associative memories. We have also tested backpropagation NNs with one hidden layer. Associative memories displayed recognition quality not worse than backpropagation networks.

1.INTRODUCTION

Chemical image (CI) recognition is identification of substance using chemical sensors' data. To solve these problems one generally use classical methods of statistical analysis such as principal component (PCA), discriminant factor analysis et c. Nowadays Feed-Forward neural networks and Kohonen's self-organizing maps are also introduced (applied)[1,2]. In this case one can reduce system tuning to training neural networks using experimental data. Unfortunately, training neural networks takes a lot of time, so their application is often discouraged. Therefore, associative memories are welcomed because of their greatly faster training (only within one iteration for Hopfield-type networks) [3]. The major disadvantage of associative memories is a rather low generalizing ability that increases sensor stability requirements. Although, new learning algorithms and architectures of associative memory enable to overcome these difficulties. In particular, modular and hetero-associative memories could be used.

In the case of piezoelectric transducers adsorption process can be directly characterized with the resonance frequency change values (at different moments) and their time derivatives, as well as various combinations of the above values. One should specially note that use of approaches based on experimental curves fitting with analytical functions to obtain the response parameters seems to be inappropriate in this case. The reason is that very often the simple kinetic models for adsorption cannot adequately describe the curves for the sensor array response, because of Multi-Component Mixture's (MCM) components diffusion into the sensitive coating bulk, change of its structure, uncontrolled fluctuation of temperature and pressure. Indeed, if one uses for *MCM CI* formation only stationary values (*i.e.*, those that characterize a system after an equilibrium has established), then one may omit important information concerning character of interaction between highly volatile components with different mobilities/activities and the sensitive layer. And the sensitive layer prehistory strongly affects the features of the receptor/analyte interaction in the initial portion of the kinetic curve. Thus the problem how to choose an optimal range of the initial data that could take into account the kinetic and stationary features of the sensor array response, as well as some leveling effects of the pre-starting procedure, are important for both fundamental science and practical applications.

The goal of this paper is to prove possibility of practical application of associative memories to chemical image recognition. We have made a series of experiments on learning to recognize some different substances (perfumes, ethanol, et c.) using associative and Feed-Forward neural networks. The research was done using neural software package NeuroLand designed at Institute of Mathematical Machines and Systems of NASU and the multi-sensor piezo-crystal gas analyzer designed at Institute of Semiconductor Physics of NASU [3].

2.EXPERIMENTAL TECHNIQUE AND GIVEN DATA

To obtain chemical images we have used an experimental assembly based on Quartz Crystal Microbalance (QCM). The universal neural computer NeuroLand was used to process digitised data: It enables to model different types of neural networks, train and test them.

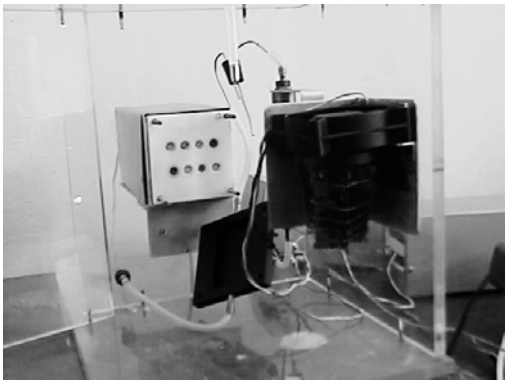


Fig. 1. Overall view of QCM experimental appliance

2.1.The QCM experimental assembly

In sensors based on piezoelectric physical transducers there is the dependence between resonance frequency *f* of acoustical resonator and mass *Δm* of a substance at sensor's surface. It is used to detect intermolecular interactions. For measurements in gas phase dependence between the resonance frequency change *Δf* and the mass change *Δm* on its surface is assumed to be linear in accordance with well-known Sauerbreu's equation.

In this work QCM-based arrays were used as prototypes an artificial nose for testing of some perfumes and vapors. For experiments a full-automatic 10-MHz quartz crystals (AT-cut, RK169) based on 8-channel systems with a measurement step of 1 s were used. The *e-Nose* instrument involves the following units: a thermostatic measuring chamber with a flow-type sensor array; a quartz oscillator unit; a frequency counter with RS232 interface based on AT89C2051 microprocessor; a gas mixtures generator; a computerized system to collect and process information. The general view of the *e-Nose* instrument is shown in Fig. 1. The sensor units have a specially designed Teflon holder in which the quartz crystals were fixed along its perimeter. So, only one side of quartz crystal was covered by sensing material and contacted with changing ambient. The flow of buffer gas (argon) was used for recovery thin film coating after adsorption phase. It was shown that construction with fixation of quartz crystals along perimeter is preferable due to strong decrease of noise.

The measuring procedure involved the following stages at the temperature 37±0.3 ^{O}C: argon circulation until the transducer frequency is stabilized (±2 Hz); vapor-gas mixture circulation at a gas-carrier rate of 20-50 ml/min.; circulation with argon air until the QCM frequency returns to its initial value.

The following five types of perfumes (GOST 17237-93) as typical MCM were tested: éau-de-Cológne "Siren", "Landish" (produced by "Effect", Kharkov); éau-de-Cológne "Russkij Les" (OAO, produced by "Kombinat Krimskaja Rosa", Simferopol), éau-de-Cológne "Shipr" and "Roksolana" (produced by PKK "ROSO", Zhovka) as well as water ant ethyl alcohol. The sample volume was 12 ml in every case.

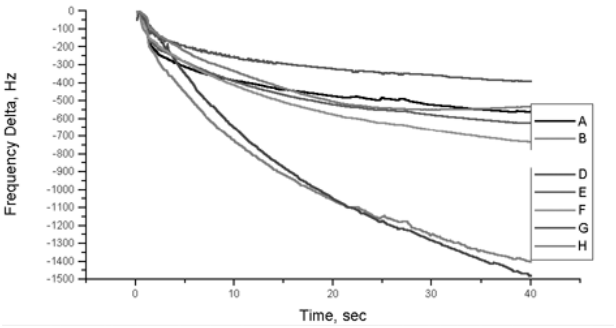


Fig. 2. Response of 7 sensors (A, B, D-H) for ethanol's smell

Table 1. Classes of chemical images

Class #	Substance	Number of images
1.	*Ethanol*	11
2.	*"Landish"*	5
3.	*"Russki Les"*	8
4.	*"Roksolana"*	6
5.	*"Shipr"*	12
6.	*"Siren"*	15
7.	*Water*	3

2.2Input data for training NN

For numerical experiments we have used measurements for seven substances (see Table 1).

Most of chemical images contained data for 7 sensors. Time series for each sensor were 286 ticks long. For instance, in the Fig. 2 we can see 7 sensors' output for the odour of ethanol.

Approximately, these regularities are close to exponential. On the other hand, a noise component is rather strong, and starting pieces of time series are sometimes instable. Note that different chemical images for each class are also highly scattered. For certain images intra-class dispersion may exceed 10%.

Data might enter into NN's input immediately, or after PCA preprocessing. Principal components were computed using the entire training sample. Otherwise NNs were trained and tested using 6 classes only (57 images, classes #1-6).

In this paper we evaluate maximum classification quality using different NNs. Also we try to find optimal

parameters of preprocessing technique. Sample window metrics (size and starting point) and number of PCs were such tuneable parameters.

The "*correctly recognized images*"/"*all test images*" ratio is called classification rate. Each value of classification rate was obtained in averaging over several "training/test" dissections of the sample. These dissections were randomly generated holding all other experiment conditions constant. To determine recognition quality we have used 50 for the modular associative classifier, and 10 dissections for the hetero-associative one. For the Feed-Forward NN recognition quality was estimated in averaging over 10 results with different initial states of the network.

3. ARCHITECTURE AND ALGORITHMS OF NEURAL NETWORKS

3.1. Associative memories

Associative memories are preferred due to fast learning done within single iteration in computing all neurons' weights. Unfortunately, most common models like Hopfield networks or bi-directional associative memories deal with bipolar data only [4]. It's needed to use convergence examination for restoration of corrupted data. But we need only classify chemical sensors' data. So, we should build an associative-memory classifier, which will use real data vectors. We have considered two associative memory classifiers: hetero-associative and a modular one.

3.1.1.Hetero-assaciative classifier

Hetero-associative classifier is a single-layered network with linear activation function; chemical sensors' data are directed to its N inputs. The quantity of neurons K is equal to number of classes (chemical substances). Weights are computed using pseudo-inverse learning rule [6]. These rules cold be obtained solving stability equation:

$$\mathbf{B}\mathbf{x}_i = \mathbf{y}_i, \quad (1)$$

where: $\mathbf{B}$ is a $K \times N$ weight matrix of thee network, $\mathbf{x_i}$ – N-dimensional input vector, $\mathbf{y_i}$ – K-dimensional output vector,its k-th component is equal to +1 if corresponding input belongs to class k, the rest of components are –1.

The training data array could be represented as two matrices: $\mathbf{X}$ contains M data vectors, and $\mathbf{Y}$ contains M desired network's responses.

Solution of (1) is:

$$\mathbf{B} = \mathbf{Y}\mathbf{X}^+, \quad (2)$$

where $\mathbf{X}^+$ is a pseudo-inverse matrix to $\mathbf{X}$.

$\mathbf{X}^+$ is usually computed using Greville's formulae [6].

3.1.2. Modular auto-associative classifier

It consists of K neural modules; each of them is a conventional auto-associative network with real stored images. Every module memorise N-dimensional data vectors for certain class. Data are memorised in k-th module computing projection matrix.

$$\mathbf{C}_k = \mathbf{X}_k \mathbf{X}_k^+, \quad (3)$$

where $\mathbf{X}_k$ is a matrix containing data vectors for k-th class.

During recognition of an input vector $\mathbf{z}$ the post-synaptic potential is computed:

$$\mathbf{s}_k = \mathbf{C}_k \mathbf{z}, \quad (4)$$

The letter is a projection of an input vector to linear span of training images for k-th class. The decision on belonging of input image is made maximizing scalar product:

$$q_k = \mathbf{s}_k \mathbf{z}^T \quad (5)$$

where $\mathbf{z}^T$ is a conjugate vector to $\mathbf{z}$.

The vector z is treated as belonging to k^* that have $q_{k^*} \to \max$.

3.2. Classifier using Feed-forward NN

Nowadays feed-forward neural networks and back-propagation learning algorithms are conventional techniques for chemical image recognition [1,2]. Strength and weaknesses of these NNs are well known; therefore we used such networks, basically, as a benchmark for associative-memory classifiers.

We have used NNs with one hidden layer. Hidden neurons had a sigmoid activation function, without a bias. Number of inputs is a dimension of input vector; for each class there is one output neuron. Data were normalized to [0, 1] range. We have tested networks with 5, 10 and 15 hidden neurons. Learning algorithm was a standard online backpropagation with learning rate 0.01 and momentum 0.15. Learning took 6000 epochs (about 10 min. of CPU time for Intel Celeron 600 MHz); so mean square error could be decreased to 10^{-2} - $5 \cdot 10^{-3}$.

4.CLASSIFICATION QUALITY DEPENDING ON SIZE OF TRAINING SAMPLE

For these series of experiments we have used raw data for all available ticks from 1st to 286th. In the case of usage of principal components the 50 highest ones were entered into classifier. Learning took about 4 sec. of CPU time for Intel Celeron 600 MHz. Results for modular and hetero-associative classifier are shown in Fig. 3.

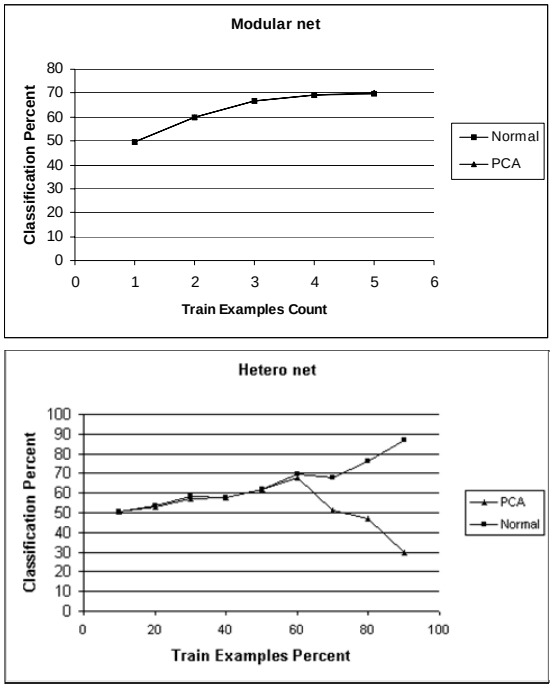


Fig. 3. Classification quality subject to training image number

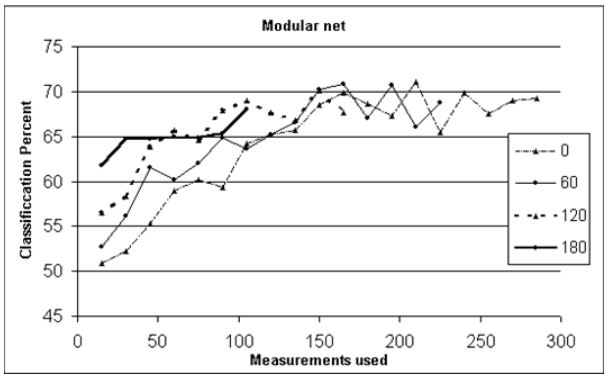


Fig. 4. Recognition quality subject to initial measurement and number of ticks used for modular network

For modular classifier the PC results are almost identical to ones for raw data. The best results – 70% – are achieved under use et least 4 learning examples. So, principal components are not strongly needed, because all useful information is extracted by NN itself.

For hetero-associative network classification quality decreases if training sample contains more than 60% of available vectors. It is caused by overflow of hetero-associative memory when number of memorized images exceeds 50% of data dimension. In this case data vectors consist of 50 components; there were 57 images, 60% of them (i.e. 34) were memorized. In a modular classifier memorizes images of one class only; so there was no overflow

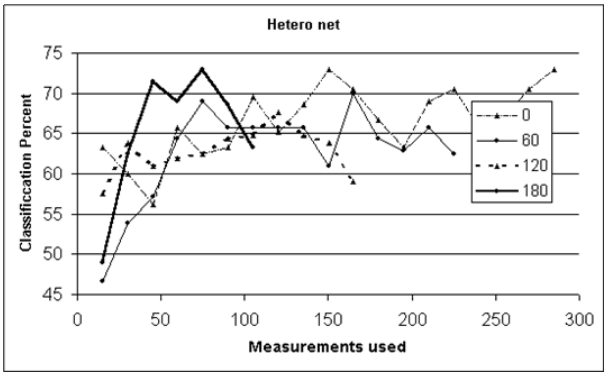


Fig. 5. Recognition quality subject to initial measurement and number of ticks used for hetero-associative network

5. DEPENDENCES ON WINDOW SIZE

These experiments were dedicated to find optimal parameters of data window for entering data into network. All types of NNs were tested. Modular network was trained using 4 examples for each class; training sample for hetero-associative one took 60% of available images. Positions of initial and final time ticks were scanned with step of 15 ticks.

Dependences of classification quality on size of data vector for associative memories are shown in the Fig. 4,5. Such dependences for Feed-forward NNs are displayed in Fig. 6. Starting point of a windows serves as a parameter in all the figures.

Note that both classifiers achieve best quality in excluding starting segment of data (100-150 ticks) and taking not more than 150-200 points for the network input. This may be explained taking into account instability of starting part of curves shown in fig 2. Note also that hetero-associative memory displays slightly better quality than modular one (73% vs. 71%). This might be caused only by experimental error.

Results for Feed-Forward networks show that initial parts of sensor response curves have low information. Best classification rates – 75-77% - were obtained excluding first 100 ticks and taking data for 50-100 ticks of the middle of response curves.

6. CLASSIFICATION RATIO SUBJECT TO NUMBER OF PRINCIPAL COMPONENTS

These experiments were dedicated to find minimal quantity of PCs required to keep admissible classification rat for associative memories. Modular network was trained using 4 examples for each class; training sample for hetero-associative one took 60% of available images. Test results are shown in Fig. 7.

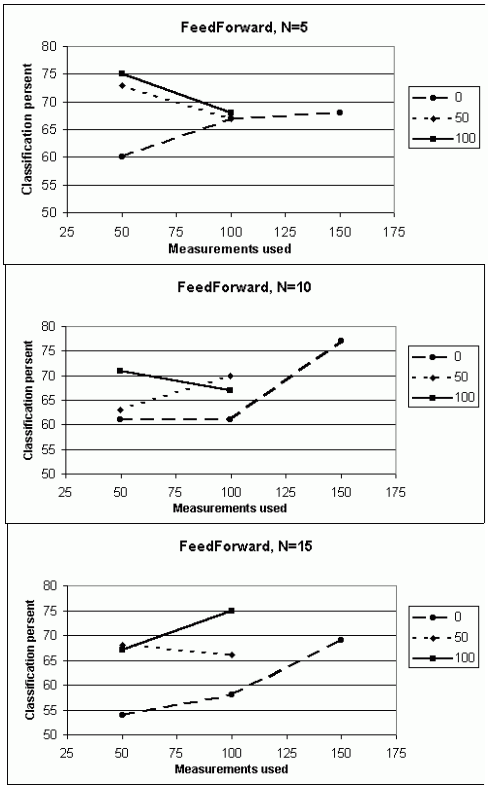


Fig. 6. Recognition quality subject to initial measurement and number of ticks used for feed forward networks with different number of hidden neurons.

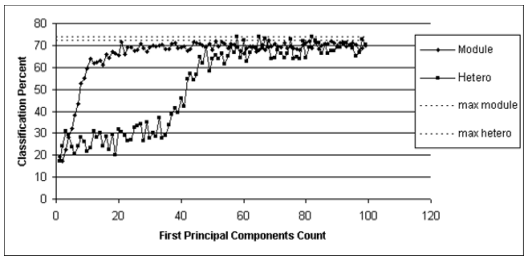


Fig. 7. Recognition quality subject to number of PCs for hetero-associative and modular NNs

Modular network displays the best classification rate if more than 25 PCs are used; hetero-associative one requires 50 PCs. It is caused by overflow of hetero-associative memory when number of memorized images exceeds 50% of data dimension (cf. Fig. 3.). We can avoid memory overflow taking well over 34 components.

7. CONCLUSION

Experimental results described above show that associative memories provide approximately the same recognition quality as feed-forward neural networks. We stress that non-iterative nature of neural associative memories makes them quite attractive. This is not only high-speed learning but also learning independent on a (random) network initialization. Even a low generalisation ability of associative memories may be turned into advantage; using more stable chemical sensors they will be able to recognise thousands sorts of chemical substance.

8. ACKNOWLEDGEMENTS

This work was partially supported by INTAS grant No. 01POLL-0257.

Calyxarenes were kindly given to us by Prof. V.I. Kal'chenko (Institute of Organic Chemistry of NASU, Kyiv). Linear polyacenes were kindly given to us by Prof. Ya.I. Vertsimakha (Institute of Physics of NASU, Kyiv). Mycrocyclic compounds were kindly given to us by Prof. Ya.D. Lampeka (Institute of Physical Chemistry of NASU, Kyiv).

9.REFERENCES

1. Keller P. E. Overview of Electronic nose Algorithms. *Proc. Internat. Joint Conf. on Neural Networks*, 1999
2. J.W. Gardner, P.N. Bartlett, *Electronic Noses. Principles and Applications*, Oxford University Press, 1999.
3. Reznik A.M "Non-Iterative Learning for Neural Networks" *Proceedings International Joint Conference on Neural Networks,* Washington DC, July 10-16, 1999
4. Kruglenko I.V., Snopok B.A., Shirshov Y.M., Venger E.F. Digital aroma technology for chemical sensing: temporal chemical images of complex mixtures // Semiconductor Physics, Quantum Electronics&Optoelectronics –2000-Vol.3-N4-P.529-541
5. Wasserman P.D., *Neural computing. Theory and practice,* ANZA Research Inc. VAN NOSTRAD REIHOLD, NY 1988
6. Kirichenko N. F, Reznik A.M, Schetenyuk S. P. "Matrix pseudo-inversion in the problem of design of associative memory", *Cybernetics and System Analysis,* vol. 37 No. 3, pp. 308-316, 2001.